Package: netmem (via r-universe)

October 24, 2026

Version 1.1-0

Date 2026-09-18

Depends R (>= 4.0.0)

Imports Matrix, stats

Suggests igraph, knitr, rmarkdown, covr, testthat, usethis, styler

Title Social Network Measures using Matrices

Maintainer Alejandro Espinosa-Rada <anespinosa@uc.cl>

Description Provides measures to describe and manipulate one-mode, two-mode, multiplex, and multilevel networks using matrix algebra. Implements functions for network centrality, cohesive subgroups, communities, structural holes, roles and positions, similarity measures, path distances, signed networks, segregation, social influence, and random network generation. Supports ego-centric and whole-network analyses, including dyadic and triadic censuses, structural balance, bipartite projections, measures with overlapping group memberships, Q-analysis, neighbourhood-inclusion dominance, main path analysis of citation networks, and permutation tests for networks. Key references: Bonacich (1972) <doi:10.1080/0022250X.1972.9989806>, Breiger (1974) <doi:10.2307/2576011>, Kivela et al. (2014) <doi:10.1093/comnet/cnu016>, Espinosa-Rada et al. (2024) <doi:10.1016/j.socnet.2023.11.008>, Schoch and Brandes (2016) <doi:10.1017/S0956792516000401>, Traag et al. (2019) <doi:10.1038/s41598-019-41695-z>, Everett and Borgatti (2026) <doi:10.1016/j.socnet.2025.12.001>.

License GPL-3

Encoding UTF-8

Language en-US

URL https://github.com/anespinosa/netmem, https://anespinosa.github.io/netmem/

BugReports https://github.com/anespinosa/netmem/issues

VignetteBuilder knitr

RoxygenNote 7.3.3

LazyData true

Repository https://anespinosa.r-universe.dev

Date/Publication 2026-09-24 11:57:18 UTC

RemoteUrl https://github.com/anespinosa/netmem

RemoteRef HEAD

RemoteSha 7bc4a983c0fca6d473f9ec60e70c1defbb979935

Contents

adj_to_incidence . . . 4
adj_to_matrix . . . 5
aggregate_layers . . . 6
alter_composition . . . 8
alter_heterogeneity . . . 9
alter_homophily . . . 10
betweenness_centrality . . . 12
block_density . . . 13
bonacich_norm . . . 15
bonacich_power . . . 16
brokerage_roles . . . 17
campnet . . . 19
centrality_centralization . . . 20
citation_decay . . . 21
clique_max . . . 22
clique_table . . . 23
closeness_centrality . . . 25
co_occurrence . . . 26
communities . . . 27
components_id . . . 29
compound_relation . . . 30
concor . . . 32
core_periphery . . . 33
cug_test . . . 34
cumulativeSumMatrices . . . 36
dag . . . 37
dir_inclusion . . . 39
dist_geographic . . . 40
dist_sim_matrix . . . 41
distances . . . 43
dominance_layers . . . 45
dominance_pairs . . . 47
dominance_ranks . . . 48
dyad_triad_table . . . 49

dyadic_census . . . 50
eb_constraint . . . 51
edgelist_to_matrix . . . 53
ego_net . . . 54
ei_index . . . 56
eigenvector_centrality . . . 57
expand_matrix . . . 58
extract_component . . . 59
FIFAego . . . 60
FIFAex . . . 61
FIFAin . . . 61
fractional_approach . . . 62
gen_degree . . . 64
gen_density . . . 65
geodesics . . . 67
heterogeneity . . . 69
hyperevent_dominance . . . 70
hypergraph . . . 72
ind_rand_matrix . . . 73
indirect_rel . . . 75
jaccard . . . 76
k_core . . . 78
katz_centrality . . . 80
kp_reciprocity . . . 81
krackhardt_friends . . . 82
krackhardt_index . . . 82
lazega_lawfirm . . . 84
leading_eigen . . . 85
leiden . . . 86
main_path . . . 87
main_path_diag . . . 89
matrix_adjlist . . . 91
matrix_projection . . . 92
matrix_report . . . 93
matrix_to_edgelist . . . 94
meta_matrix . . . 95
minmax_overlap . . . 97
mix_matrix . . . 98
mixed_census . . . 99
modularity_score . . . 100
multilevel_degree . . . 102
multiplex_census . . . 105
neigh_inclusion . . . 107
page_rank_centrality . . . 109
pareto_dominance . . . 110
partition_centrality . . . 112
percolation_clique . . . 113
perm_label . . . 114

perm_matrix . . . 115
pos_dominance . . . 116
posneg_index . . . 117
power_function . . . 118
pref_attachment . . . 119
preserved_order . . . 120
q_analysis . . . 121
qap_cor . . . 124
qap_lm . . . 126
recip_coef . . . 127
redundancy . . . 128
rege . . . 129
segregation . . . 130
set_inclusion . . . 132
shared_partners . . . 133
simplicial_complexes . . . 134
small_world . . . 136
social_influence . . . 137
spatial_cor . . . 139
struc_balance . . . 140
structural_holes . . . 141
structural_na . . . 143
supra_adjacency . . . 144
threshold_diffusion . . . 145
trans_coef . . . 147
trans_matrix . . . 148
traversal_weights . . . 149
triad_uman . . . 152
z_arctest . . . 154
zone_sample . . . 154

Index **157**

`adj_to_incidence` *Convert an Adjacency Matrix to an Incidence Matrix*

Description

This function transforms an adjacency matrix into an incidence matrix.

Usage

```
adj_to_incidence(A, loops = TRUE, directed = TRUE, weighted = TRUE)
```

Arguments

A	A square numeric matrix representing the adjacency matrix of a graph. The matrix should have non-negative values, where 'A[i, j]' represents the weight of the edge from node 'i' to node 'j'.
loops	Logical. If 'TRUE', self-loops (edges from a node to itself) are included in the incidence matrix. If 'FALSE', they are removed. Default is 'TRUE'.
directed	Logical. If 'TRUE', the graph is treated as directed, meaning each edge has a specific source and target. If 'FALSE', the graph is treated as undirected, and edges are symmetrically represented. Default is 'TRUE'.
weighted	Logical. If 'TRUE', edge weights from 'A' are included in the incidence matrix. If 'FALSE', all edges are treated as having weight '1'. Default is 'TRUE'.

Value

A numeric matrix where rows represent nodes and columns represent edges. - In a **directed** network, a source node has a negative value (-weight), and a target node has a positive value (+weight). - In an **undirected** network, both nodes involved in an edge share the weight (positive values). - If 'weighted = FALSE', all edges have a weight of '1'.

Examples

```
# Define an adjacency matrix (directed and weighted)
A <- matrix(c(
  1, 3, 0, 0, 2,
  0, 0, 2, 0, 0,
  5, 0, 0, 0, 0,
  0, 0, 0, 0, 1,
  0, 4, 0, 0, 0
), byrow = TRUE, nrow = 5)

# Convert to an incidence matrix (directed, weighted)
(inc_matrix <- adj_to_incidence(A))

# Undirected, weighted graph
(inc_matrix_undirected <- adj_to_incidence(A, directed = FALSE))

# Directed, unweighted graph
(inc_matrix_unweighted <- adj_to_incidence(A, weighted = FALSE))

# Ignore loops
(inc_matrix_no_loops <- adj_to_incidence(A, loops = FALSE))
```

---

adj_to_matrix *Transform an adjacency list into a matrix*

Description

Transform an adjacency list into a matrix

Usage

```
adj_to_matrix(A, type = c("adjacency", "incidence", "weighted"), loops = FALSE)
```

Arguments

A	An adjacent list
type	Transform the adjacent list into an adjacency matrix, an incidence matrix or a weighted matrix
loops	Whether to include loops into the matrix

Value

This function transforms an adjacency list into a matrix

Author(s)

Alejandro Espinosa-Rada

Examples

```
adj_groups <- rbind(
  c("a", "b", "c"), c("a", "c", NA),
  c("b", "c", NA), c("c", NA, NA),
  c("c", "a", NA)
)
M <- adj_to_matrix(adj_groups, type = "adjacency", loops = TRUE)
M
```

---

aggregate_layers *Aggregation of the layers*

Description

Reduces a multiplex network to a single matrix, in which the actors are joined when they are connected in the layers (Battiston et al., 2014; De Domenico et al., 2015).

Usage

```
aggregate_layers(layers, method = c("sum", "binary", "mean"), l = NULL)
```

Arguments

layers	A list of square matrices of the same order, or a supra-adjacency matrix with l layers
method	Whether the ties are added (sum, default), made binary (binary) or averaged (mean)
l	The number of layers, when layers is a supra-adjacency matrix that does not come from supra_adjacency()

Details

method = "sum" (default) adds the layers, so the value of a tie is the number of layers in which it is present, which Battiston et al. (2014) call the overlapping network. "binary" gives one to every tie that is present in at least one layer, and "mean" divides the sum by the number of layers.

The layers are given as a list of matrices, or as the supra-adjacency matrix of supra_adjacency() together with the number of layers l, in which case the coupling between the layers is ignored.

Aggregating loses the information of which layer each tie belongs to, and measures computed on the aggregated network can differ from the ones computed on the layers (De Domenico et al., 2015).

Value

This function returns a square matrix with the actors of the layers.

Author(s)

Alejandro Espinosa-Rada

References

Battiston, F., Nicosia, V. and Latora, V. (2014). Structural measures for multiplex networks. Physical Review E, 89(3), 032804. doi:10.1103/PhysRevE.89.032804

De Domenico, M., Nicosia, V., Arenas, A. and Latora, V. (2015). Structural reducibility of multilayer networks. Nature Communications, 6, 6864. doi:10.1038/ncomms7864

Examples

```
A1 <- matrix(c(
  0, 1, 0,
  1, 0, 1,
  0, 1, 0
), byrow = TRUE, ncol = 3, dimnames = list(letters[1:3], letters[1:3]))

A2 <- matrix(c(
  0, 1, 1,
  1, 0, 0,
  1, 0, 0
), byrow = TRUE, ncol = 3, dimnames = list(letters[1:3], letters[1:3]))

# The tie a-b is in both layers
aggregate_layers(list(A1, A2))
```

```
aggregate_layers(list(A1, A2), method = "binary")

# The same from the supra-adjacency matrix, which knows its layers
aggregate_layers(supra_adjacency(list(A1, A2)))
```

`alter_composition` *Alter composition*

Description

Number of alters of each node in each category, for categories that can overlap (Everett and Borgatti, 2026).

Usage

```
alter_composition(A, B, proportion = FALSE)
```

Arguments

`A`	A matrix of the ties of the nodes (rows) with their alters (columns)
`B`	A matrix of the membership of the alters (rows) in the categories (columns), or a vector with the category of each alter
`proportion`	Whether to divide each row by its sum, giving the proportion of the alters in each category

Details

The memberships `B` are made row-stochastic, so that each row gives the proportion of a node that belongs to each category (for instance, the proportion of time spent on each project). The alter composition is the product AB: for a partition, it counts the alters of each node in each category; with overlapping categories, it gives the extent to which the alters of each node belong to each category. The rows of AB add up to the degree of each node. `A` need not be square: its columns should be the rows of `B`, as in a matrix of respondents and the alters they named.

Value

This function returns a matrix of the nodes (rows) and the categories (columns).

Author(s)

Alejandro Espinosa-Rada

References

Everett, M. G. and Borgatti, S. P. (2026). Alter composition with overlapping group memberships. Social Networks, 85, 80–88. doi:10.1016/j.socnet.2025.12.001

Examples

```
A <- matrix(c(
  0, 1, 1, 0,
  1, 0, 1, 1,
  1, 1, 0, 0,
  0, 1, 0, 0
), byrow = TRUE, ncol = 4)
rownames(A) <- colnames(A) <- c("a", "b", "c", "d")

# A partition
alter_composition(A, c("x", "x", "y", "y"))

# Overlapping categories: hours spent on two projects
B <- matrix(c(
  10, 0,
  5, 5,
  0, 8,
  2, 6
), byrow = TRUE, ncol = 2)
alter_composition(A, B)
```

---

alter_heterogeneity *Alter heterogeneity*

Description

Blau's heterogeneity of the alters of each node, for categories that can overlap (Everett and Borgatti, 2026).

Usage

```
alter_heterogeneity(A, B, normalized = FALSE)
```

Arguments

A	A matrix of the ties of the nodes (rows) with their alters (columns)
B	A matrix of the membership of the alters (rows) in the categories (columns), or a vector with the category of each alter
normalized	Whether to return the IQV

Details

The heterogeneity of a node is $1 - \sum_k p_k^2$, where p_k is the proportion of its alters in category k, taken from the rows of `alter_composition()` (Blau, 1977). A node with a single alter has some heterogeneity when that alter belongs to several categories. The IQV divides the index by its maximum, $1 - 1/K$, where K is the number of categories. The nodes without alters are `NA`.

Value

This function returns a vector with the heterogeneity of the alters of each node.

Author(s)

Alejandro Espinosa-Rada

References

Blau, P. M. (1977). Inequality and heterogeneity: A primitive theory of social structure. Free Press.

Everett, M. G. and Borgatti, S. P. (2026). Alter composition with overlapping group memberships. Social Networks, 85, 80–88. doi:10.1016/j.socnet.2025.12.001

Examples

```
A <- matrix(c(
  0, 1, 1, 0,
  1, 0, 1, 1,
  1, 1, 0, 0,
  0, 1, 0, 0
), byrow = TRUE, ncol = 4)
rownames(A) <- colnames(A) <- c("a", "b", "c", "d")
B <- matrix(c(
  10, 0,
  5, 5,
  0, 8,
  2, 6
), byrow = TRUE, ncol = 2)

alter_heterogeneity(A, B)
```

---

alter_homophily *Alter homophily*

Description

E-I index and Yule's Q of each node, for categories that can overlap (Everett and Borgatti, 2026).

Usage

```
alter_homophily(
  A,
  B,
  method = c("ei", "yule"),
  similarity = c("product", "minimum", "cosine")
)
```

Arguments

A	A square matrix
B	A matrix of the membership of the nodes (rows) in the categories (columns), or a vector with the category of each node
method	The index: ei (default) or yule
similarity	The similarity of the memberships: product (default), minimum or cosine

Details

The internal ties of a node i are $I = \sum_j A_{ij} S_{ij}$, where S_{ij} is the similarity of the memberships of i and j, and the external ties are $E = D - I$, where D is the degree. The E-I index is $(E-I)/(E+I)$ (Krackhardt and Stern, 1988): -1 when all the alters are in the categories of ego (homophily) and +1 when none is (heterophily).

Yule's Q also uses the nodes that are not alters, to take into account how many nodes of each category are available: $a = I$ and $b = E$ for the alters, and c and d are the same quantities for the other nodes. Then $Q = (ad - bc)/(ad + bc)$, which is positive for homophily, and zero when the ties do not depend on the categories.

The similarity of two memberships can be defined in three ways:

`similarity = "product"` (default), $S_{ij} = \sum_k B_{ik} B_{jk}$. It reduces to the usual indices when the categories are a partition. If the categories are, for instance, the proportion of time spent in each of several places, it is the probability that two nodes are in the same place.

`similarity = "minimum"`, $S_{ij} = \sum_k \min(B_{ik}, B_{jk})$, the trait version ($E_s - I_s$ and Q_s). Two nodes with identical memberships are fully similar even when they split their time among categories, which fits categories that are traits such as skills or interests.

`similarity = "cosine"`, the cosine of the memberships, which favours two nodes that concentrate in the same categories, such as two specialists with the same specialty.

The ties are binary and the loops are ignored. For a directed network the alters are the out-neighbours. Nodes without alters, and Yule's Q with $ad + bc = 0$, are NA.

Value

This function returns a vector with the index of each node.

Author(s)

Alejandro Espinosa-Rada

References

Everett, M. G. and Borgatti, S. P. (2026). Alter composition with overlapping group memberships. Social Networks, 85, 80–88. doi:10.1016/j.socnet.2025.12.001

Krackhardt, D. and Stern, R. N. (1988). Informal networks and organizational crises: An experimental simulation. Social Psychology Quarterly, 51(2), 123–140. doi:10.2307/2786835

Examples

```
A <- matrix(c(
  0, 1, 1, 0,
  1, 0, 1, 1,
  1, 1, 0, 0,
  0, 1, 0, 0
), byrow = TRUE, ncol = 4)
rownames(A) <- colnames(A) <- c("a", "b", "c", "d")
B <- matrix(c(
  10, 0,
  5, 5,
  0, 8,
  2, 6
), byrow = TRUE, ncol = 2)

alter_homophily(A, B)
alter_homophily(A, B, method = "yule", similarity = "minimum")
```

---

betweenness_centrality

Betweenness centrality

Description

Betweenness centrality of Freeman (1977), computed with the algorithm of Brandes (2001).

Usage

```
betweenness_centrality(
  A,
  digraph = TRUE,
  weighted = FALSE,
  alpha = 1,
  normalized = FALSE
)
```

Arguments

A	A square matrix
digraph	Whether the matrix is directed or undirected
weighted	Whether the matrix is weighted
alpha	The tuning parameter of Opsahl et al. (2010) to transform weights into lengths
normalized	If TRUE, the result is divided by (n-1)(n-2) for directed networks and (n-1)(n-2)/2 for undirected networks

Details

The betweenness of a node is the sum, over all pairs of other nodes, of the proportion of geodesics between the pair that pass through the node. For undirected networks each pair is counted once.

For valued matrices, the tie weights are treated as strengths and transformed into lengths as $1/w^{\alpha}$ (Opsahl et al., 2010). If `alpha = 0` the binary network is used.

Value

This function returns the betweenness centrality of the nodes.

Author(s)

Alejandro Espinosa-Rada

References

Brandes, U. (2001). A faster algorithm for betweenness centrality. Journal of Mathematical Sociology, 25(2), 163–177. doi:10.1080/0022250X.2001.9990249

Freeman, L. C. (1977). A set of measures of centrality based on betweenness. Sociometry, 40(1), 35–41. doi:10.2307/3033543

Opsahl, T., Agneessens, F., and Skvoretz, J. (2010). Node centrality in weighted networks: Generalizing degree and shortest paths. Social Networks, 32(3), 245–251. doi:10.1016/j.socnet.2010.03.006

Examples

```
A <- matrix(c(
  0, 1, 1, 1, 0, 0, 0, 0, 0,
  1, 0, 1, 1, 1, 0, 0, 0, 0,
  1, 1, 0, 1, 0, 1, 0, 0, 0,
  1, 1, 1, 0, 1, 1, 0, 0, 0,
  0, 1, 0, 1, 0, 1, 1, 0, 0,
  0, 0, 1, 1, 1, 0, 1, 0, 0,
  0, 0, 0, 0, 1, 1, 0, 1, 0,
  0, 0, 0, 0, 0, 0, 1, 0, 1,
  0, 0, 0, 0, 0, 0, 0, 1, 0
), byrow = TRUE, ncol = 9)
rownames(A) <- letters[1:nrow(A)]
colnames(A) <- rownames(A)

betweenness_centrality(A, digraph = FALSE)
```

---

block_density *Block densities and image matrix*

Description

Densities of the blocks given by a partition of the nodes, and the image matrix that summarises them (Lorrain and White, 1971; Wasserman and Faust, 1994).

Usage

```
block_density(A, partition, cutoff = "density", loops = FALSE)
```

Arguments

A	A square matrix
partition	A vector with the position of each node
cutoff	Density above which a block is a one in the image matrix: a number, or density (default) for the density of the whole network
loops	Whether the loops are counted as possible ties in the diagonal blocks

Details

A partition of the nodes divides the matrix into blocks. The density of a block is the proportion of the possible ties that are present, and the image matrix assigns a one to the blocks whose density is at least the cutoff. The usual criterion is the density of the whole network (cutoff = "density"), so a block is one when it is denser than the network as a whole.

The diagonal blocks contain the ties within a position, where the possible ties exclude the loops unless loops = TRUE.

Value

This function returns the density of each block, the image matrix and the density of the network.

Author(s)

Alejandro Espinosa-Rada

References

Lorrain, F. and White, H. C. (1971). Structural equivalence of individuals in social networks. Journal of Mathematical Sociology, 1(1), 49–80. doi:10.1080/0022250X.1971.9989788

Wasserman, S. and Faust, K. (1994). Social network analysis: Methods and applications. Cambridge University Press.

Examples

```
A <- matrix(c(
  0, 1, 1, 0, 0, 0,
  1, 0, 1, 0, 0, 0,
  1, 1, 0, 1, 1, 1,
  0, 0, 1, 0, 0, 0,
  0, 0, 1, 0, 0, 0,
  0, 0, 1, 0, 0, 0
), byrow = TRUE, ncol = 6)
rownames(A) <- letters[1:nrow(A)]
colnames(A) <- rownames(A)

block_density(A, partition = c(1, 1, 1, 2, 2, 2))
```

bonacich_norm *Bonacich normalization*

Description

The function provide a normalisation provided by Bonacich (1972).

Usage

```
bonacich_norm(A, projection = c("rows", "columns"), normalisation = FALSE)
```

Arguments

A	An incidence matrix
projection	Whether to normalise by rows (default), or columns of the matrix.
normalisation	Normalise the measure

Value

This function returns the Bonacich normalisation.

Source

Adapted from Borgatti, S., Everett, M., Johnson, J. and Agneessens, P. (2022) Analyzing Social Networks Using R. Sage.

References

Bonacich, P. (1972). Factoring and weighting approaches to status scores and clique identification. Journal of Mathematical Sociology, 2: 112-120.

Examples

```
A <- matrix(
  c(
    1, 1, 1, 1, 1, 1, 0, 1, 1, 0, 0, 0, 0, 0,
    1, 1, 1, 0, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0,
    0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0,
    1, 0, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0,
    0, 0, 1, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0,
    0, 0, 1, 0, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0,
    0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0,
    0, 0, 0, 0, 0, 1, 0, 1, 1, 0, 0, 0, 0, 0,
    0, 0, 0, 0, 1, 0, 1, 1, 1, 0, 0, 0, 0, 0,
    0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 1, 0, 0,
    0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 1, 0, 0,
    0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 1, 1, 1,
    0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 0, 1, 1, 1,
    0, 0, 0, 0, 0, 1, 1, 0, 1, 1, 1, 1, 1, 1,
```

```
    0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 1, 1, 0, 0,
    0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0,
    0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0,
    0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0
  ),
  byrow = TRUE, ncol = 14
)
bonacich_norm(A)
```

`bonacich_power` *Bonacich power centrality*

Description

Power centrality of Bonacich (1987), where being connected to well-connected others can increase or decrease the centrality of a node.

Usage

```
bonacich_power(
  A,
  beta = 0,
  digraph = TRUE,
  weighted = FALSE,
  scale = c("none", "ssq")
)
```

Arguments

`A`	A square matrix
`beta`	The weight given to the centrality of the neighbours. It should be smaller than the inverse of the leading eigenvalue
`digraph`	Whether the matrix is directed or undirected
`weighted`	Whether the matrix is weighted
`scale`	Whether the scores are returned as they are (`none`, default) or scaled so that the sum of their squares is the number of nodes (`ssq`), as in other packages

Details

The centrality is $x = (I - \beta A)^{-1} A 1$. When `beta` is positive, a node is central when it is connected to central nodes, as in the eigenvector centrality. When `beta` is negative, being connected to well-connected others reduces the centrality of a node, which describes bargaining situations. With `beta = 0` the measure is the degree.

Value

This function returns the power centrality of the nodes.

Author(s)

Alejandro Espinosa-Rada

References

Bonacich, P. (1987). Power and centrality: A family of measures. American Journal of Sociology, 92(5), 1170–1182. doi:10.1086/228631

Examples

```
A <- matrix(c(
  0, 1, 1, 1, 0,
  1, 0, 0, 0, 0,
  1, 0, 0, 0, 1,
  1, 0, 0, 0, 0,
  0, 0, 1, 0, 0
), byrow = TRUE, ncol = 5)
rownames(A) <- letters[1:nrow(A)]
colnames(A) <- rownames(A)

bonacich_power(A, beta = 0.1, digraph = FALSE)
bonacich_power(A, beta = -0.1, digraph = FALSE)
```

---

brokerage_roles *Brokerage roles*

Description

Brokerage roles of Gould and Fernandez (1989), for categories that can overlap (Everett and Borgatti, 2026).

Usage

```
brokerage_roles(A, B)
```

Arguments

A	A square matrix
B	A matrix of the membership of the nodes (rows) in the categories (columns), or a vector with the category of each node

Details

A node b brokers in a path $a \to b \to c$ when a has no tie to c. The role depends on the categories of the three nodes: `coordinator` (all in the same category), `gatekeeper` (b and c in the same category, a in another), `representative` (a and b in the same category, c in another), `consultant` (also called itinerant; a and c in the same category, b in another) and `liaison` (the three in different categories).

With overlapping categories, each path counts partly for each role (Everett and Borgatti, 2026: Eq. 5). The coordinator part is $\sum_j B_{aj} B_{bj} B_{cj}$, the joint membership of the three nodes in each category; the gatekeeper part is $\sum_j B_{bj} B_{cj} (1 - B_{aj})$; the representative part $\sum_j B_{aj} B_{bj} (1 - B_{cj})$; the consultant part $\sum_j B_{aj} B_{cj} (1 - B_{bj})$; and the liaison part is the rest. When the categories are a partition, each path counts for a single role, as in Gould and Fernandez (1989). The parts of each path add up to one, so the total of each node is the number of paths it brokers.

A symmetric matrix is treated as a directed network with ties in both directions, so each path is counted from both ends.

Value

This function returns a matrix with the score of each node (rows) in each role, and the total.

Author(s)

Alejandro Espinosa-Rada

References

Everett, M. G. and Borgatti, S. P. (2026). Alter composition with overlapping group memberships. Social Networks, 85, 80–88. doi:10.1016/j.socnet.2025.12.001

Gould, R. V. and Fernandez, R. M. (1989). Structures of mediation: A formal approach to brokerage in transaction networks. Sociological Methodology, 19, 89–126. doi:10.2307/270949

Examples

```
A <- matrix(c(
  0, 1, 0, 0,
  0, 0, 1, 1,
  0, 0, 0, 1,
  1, 0, 0, 0
), byrow = TRUE, ncol = 4)
rownames(A) <- colnames(A) <- c("a", "b", "c", "d")

brokerage_roles(A, c("x", "x", "y", "y"))

B <- matrix(c(
  10, 0,
  5, 5,
  0, 8,
  2, 6
), byrow = TRUE, ncol = 2)
brokerage_roles(A, B)
```

`campnet` *Camp 92 network*

Description

Interactions among the 18 people of a three-week course, the 1992 NSF Summer Institute on Research Methods in Cultural Anthropology, 14 participants and 4 instructors. The data were collected by Steve Borgatti, Russ Bernard, Bert Pelto and Gery Ryan. At the end of the second week, each person sorted cards with the names of the others by how much they had interacted with them since the beginning of the course.

Usage

```
data(campnet)
```

Format

A list with:

network An 18 x 18 binary matrix, known as Campnet, where `network[i, j]` = 1 if `j` is among the three people with whom `i` interacted most.

ranks An 18 x 18 matrix with the rank that each respondent (row) gave to each of the others (1 = most interaction, 17 = least).

attributes A data frame with the attributes of the people:

gender 1 = woman; 2 = man

role 1 = participant; 2 = instructor

Details

The network gives the betweenness scores reported by Everett and Borgatti (2026: Table 5).

Source

UCINET datasets, Camp 92 (https://sites.google.com/site/ucinetsoftware/datasets/camp-92).

Borgatti, S. P., Everett, M. G. and Johnson, J. C. (2018). Analyzing Social Networks. Second edition. SAGE.

Everett, M. G. and Borgatti, S. P. (2026). Alter composition with overlapping group memberships. Social Networks, 85, 80–88. doi:10.1016/j.socnet.2025.12.001

`centrality_centralization` *Centralization*

Description

Centralization of Freeman (1979): how much a network is dominated by its most central node.

Usage

```
centrality_centralization(
  A,
  measure = c("degree", "closeness", "betweenness", "eigenvector"),
  digraph = FALSE,
  type = c("out", "in", "all"),
  loops = FALSE
)
```

Arguments

`A`	A square matrix
`measure`	The centrality to be used: `degree` (default), `closeness`, `betweenness` or `eigenvector`
`digraph`	Whether the matrix is directed or undirected
`type`	Type of degree or distances for directed networks
`loops`	Whether to consider the loops of the matrix

Details

The centralization is the sum of the differences between the highest centrality and the centrality of every node, divided by the largest sum that a network of the same order can have. For degree, closeness and betweenness the maximum is given by the star network, and for the eigenvector centrality by a network with a single tie, which gives $n - 2$.

The function is named `centrality_centralization()` so that it does not mask `sna::centralization()`.

Value

This function returns the centralization of the network, the centrality scores and the theoretical maximum.

Author(s)

Alejandro Espinosa-Rada

References

Freeman, L. C. (1979). Centrality in social networks conceptual clarification. Social Networks, 1(3), 215–239. doi:10.1016/03788733(78)900217

Examples

```
A <- matrix(c(
  0, 1, 1, 1, 0,
  1, 0, 0, 0, 0,
  1, 0, 0, 0, 1,
  1, 0, 0, 0, 0,
  0, 0, 1, 0, 0
), byrow = TRUE, ncol = 5)
rownames(A) <- letters[1:nrow(A)]
colnames(A) <- rownames(A)

centrality_centralization(A, measure = "degree", digraph = FALSE)
centrality_centralization(A, measure = "betweenness", digraph = FALSE)
```

---

citation_decay *Temporal decay weighting for citation edges*

Description

Applies an exponential temporal decay to the arcs of a citation adjacency matrix and optionally normalises the result so that each citing paper distributes exactly one unit of citation influence among its references.

Usage

```
citation_decay(A, years, lambda = 0.2, normalize = TRUE)
```

Arguments

A	A square, named, directed adjacency matrix.
years	Named numeric vector of publication years aligned with row/column names of A.
lambda	Positive numeric decay rate. Default 0.2.
normalize	Logical; if TRUE (default) each column is divided by its sum so that the references of each citing paper sum to one.

Details

As in traversal_weights, A[i,j] > 0 means that paper j cites paper i. The weight of the arc $i \to j$ is $\exp(-\lambda \cdot (y_j - y_i))$ where y_i and y_j are the publication years of i and j respectively. A larger λ discounts older citations more aggressively. The normalisation divides each column, which holds the references of a citing paper, by its sum, so that differences in the length of the reference lists do not inflate the raw weights.

Value

A numeric matrix of the same dimensions as `A` containing the decay-weighted (and optionally normalised) citation arc weights. `traversal_weights` uses only whether each arc is present, so these weights do not change the search path counts; they can be combined with them, for instance by multiplying the two matrices element by element.

Author(s)

Alejandro Espinosa-Rada

References

Hummon, N.P. and Doreian, P. (1989). Connectivity in a citation network: The development of DNA theory. Social Networks. 11(1): 39-63. doi:10.1016/03788733(89)900178.

Examples

```
# P1 is cited by P2 and P3, which are both cited by P4
A <- matrix(c(
  0, 1, 1, 0,
  0, 0, 0, 1,
  0, 0, 0, 1,
  0, 0, 0, 0
), byrow = TRUE, nrow = 4)
rownames(A) <- c("P1", "P2", "P3", "P4")
colnames(A) <- c("P1", "P2", "P3", "P4")
years <- c(P1 = 2000, P2 = 2005, P3 = 2006, P4 = 2010)

citation_decay(A, years, lambda = 0.2)
```

---

clique_max *Maximal cliques*

Description

Maximal complete subgraphs of an undirected network, found with the algorithm of Bron and Kerbosch (1973).

Usage

```
clique_max(A, min = 2, max = NULL)
```

Arguments

`A`	A symmetric matrix object
`min`	Minimum size of the cliques returned
`max`	Maximum size of the cliques returned. If NULL, there is no limit

Details

A clique is a set of nodes that are all adjacent to each other, and it is maximal when no other node can be added to it. Unlike `clique_table()`, which returns the triangles of the network, this function returns cliques of any size.

Value

This function returns a list with the names of the nodes of each maximal clique.

Author(s)

Alejandro Espinosa-Rada

References

Bron, C. and Kerbosch, J. (1973). Algorithm 457: Finding all cliques of an undirected graph. Communications of the ACM, 16(9), 575–577. doi:10.1145/362342.362367

Luce, R. D. and Perry, A. D. (1949). A method of matrix analysis of group structure. Psychometrika, 14(2), 95–116. doi:10.1007/BF02289146

Examples

```
A <- matrix(c(
  0, 1, 1, 0, 0, 0,
  1, 0, 1, 1, 0, 0,
  1, 1, 0, 1, 0, 0,
  0, 1, 1, 0, 1, 1,
  0, 0, 0, 1, 0, 1,
  0, 0, 0, 1, 1, 0
), byrow = TRUE, ncol = 6)
rownames(A) <- letters[1:nrow(A)]
colnames(A) <- rownames(A)

clique_max(A)
```

---

clique_table *Clique table*

Description

Exploration of a 3-cliques, as the maximum number of three or more actors who have all possible ties present among themselves

Usage

```
clique_table(A, list_cliques = FALSE, number = FALSE)
```

Arguments

A	A symmetric matrix object.
list_cliques	Whether to return the list of cliques.
number	Number of triangles

Value

This function return an edge list of actors participating in 3-cliques.

If list_cliques = TRUE it also return the list of cliques per nodes. If number = TRUE the output returns the number of 3-cliques in the matrix.

Author(s)

Alejandro Espinosa-Rada

References

Luce, R.D. and Perry, A.D. (1949). A method of matrix analysis of group structure. Psychometrika, 14: 95-116.

Roethlisberger, F.J. and Dickson, W.J. (1939). Management and the Worker. Harvard University Press, Cambridge, MA.

Wasserman, S. and Faust, K. (1994). Social network analysis: Methods and applications. Cambridge University Press.

Examples

```
A <- matrix(c(
  0, 1, 1, 0, 0, 0, 0, 1, 0,
  1, 0, 1, 0, 0, 0, 0, 0, 0,
  1, 1, 0, 0, 0, 0, 0, 0, 0,
  0, 0, 0, 0, 1, 1, 0, 0, 0,
  0, 0, 0, 1, 0, 0, 0, 0, 0,
  0, 0, 0, 1, 0, 0, 1, 1, 0,
  0, 0, 0, 0, 0, 1, 0, 1, 0,
  1, 0, 0, 0, 0, 1, 1, 0, 0,
  0, 0, 0, 0, 0, 0, 0, 0, 0
), byrow = TRUE, ncol = 9)
rownames(A) <- letters[1:nrow(A)]
colnames(A) <- rownames(A)
clique_table(A, list_cliques = TRUE, number = TRUE)
```

`closeness_centrality` *Closeness centrality*

Description

Closeness centrality of Freeman (1978) and its harmonic version (Marchiori and Latora, 2000; Rochat, 2009).

Usage

```
closeness_centrality(
  A,
  digraph = TRUE,
  type = c("out", "in", "all"),
  weighted = FALSE,
  alpha = 1,
  harmonic = FALSE,
  normalized = FALSE
)
```

Arguments

`A`	A square matrix
`digraph`	Whether the matrix is directed or undirected
`type`	Whether to use the `out` (default), `in` or `all` distances. The `all` option uses the underlying graph
`weighted`	Whether the matrix is weighted
`alpha`	The tuning parameter of Opsahl et al. (2010) to transform weights into lengths
`harmonic`	Whether to return the harmonic closeness
`normalized`	If TRUE, Freeman's closeness is multiplied by the number of nodes reached, and the harmonic closeness is divided by (n-1)

Details

Freeman's closeness is the inverse of the sum of the geodesic distances from a node to the others. When the network is disconnected, the sum only considers the nodes that can be reached, and the harmonic version is recommended, as it adds the inverse of each distance (an unreachable node adds zero).

For valued matrices, the tie weights are treated as strengths and transformed into lengths as $1/w^{\alpha}$ (Opsahl et al., 2010). If `alpha = 0` the binary network is used.

Value

This function returns the closeness centrality of the nodes.

Author(s)

Alejandro Espinosa-Rada

References

Freeman, L. C. (1978). Centrality in social networks conceptual clarification. Social Networks, 1(3), 215–239. doi:10.1016/03788733(78)900217

Marchiori, M. and Latora, V. (2000). Harmony in the small-world. Physica A, 285(3-4), 539–546. doi:10.1016/S03784371(00)003113

Opsahl, T., Agneessens, F., and Skvoretz, J. (2010). Node centrality in weighted networks: Generalizing degree and shortest paths. Social Networks, 32(3), 245–251. doi:10.1016/j.socnet.2010.03.006

Rochat, Y. (2009). Closeness centrality extended to unconnected graphs: The harmonic centrality index. ASNA.

Examples

```
A <- matrix(c(
  0, 1, 1, 1, 0,
  1, 0, 0, 0, 0,
  1, 0, 0, 0, 1,
  1, 0, 0, 0, 0,
  0, 0, 1, 0, 0
), byrow = TRUE, ncol = 5)
rownames(A) <- letters[1:nrow(A)]
colnames(A) <- rownames(A)

closeness_centrality(A, digraph = FALSE)
closeness_centrality(A, digraph = FALSE, harmonic = TRUE)
```

---

co_occurrence *Co-occurrence*

Description

Co-occurrence matrix based on overlap function

Usage

```
co_occurrence(
  A,
  similarity = c("ochiai", "cosine"),
  occurrence = TRUE,
  projection = FALSE
)
```

Arguments

A	A matrix
similarity	The similarities available are either Ochiai (default) or cosine.
occurrence	Whether to treat the matrix as a two-mode structure (a.k.a. rectangular matrix, occurrence matrix, affiliation matrix, bipartite network)
projection	Whether to apply a projection (inner product multiplication) to the matrix

Value

This function returns the normalisation of a matrix into a symmetrical co-occurrence matrix

Author(s)

Alejandro Espinosa-Rada

References

Borgatti, S. P., Halgin, D. S., 2011. Analyzing affiliation networks. In: J. Scott and P. J. Carrington (Eds.) The Sage handbook of social network analysis (pp. 417-433), Sage.

Zhou, Q., & Leydesdorff, L. (2016). The normalization of occurrence and Co-occurrence matrices in bibliometrics using Cosine similarities and Ochiai coefficients. Journal of the Association for Information Science and Technology, 67(11), 2805–2814. doi:10.1002/asi.23603

Examples

```
A <- matrix(
  c(
    2, 0, 2,
    1, 1, 0,
    0, 3, 3,
    0, 2, 2,
    0, 0, 1
  ),
  nrow = 5, byrow = TRUE
)

co_occurrence(A)
```

---

communities *Communities with agglomeration, label propagation or edge betweenness*

Description

Three classic ways of finding communities.

Usage

```
community_greedy(A, weighted = FALSE)

community_label(A, weighted = FALSE, max_iter = 100)

community_betweenness(A, weighted = FALSE)
```

Arguments

A	A symmetric matrix object
weighted	Whether the matrix is weighted
max_iter	Maximum number of rounds of label propagation

Details

community_greedy starts with every node alone and merges at each step the two groups that increase the modularity the most, keeping the partition with the highest modularity (Clauset, Newman and Moore, 2004).

community_label gives every node a different label, and then each node takes the label that most of its neighbours have, until no label changes. It is fast but the result depends on the order in which the nodes are visited, so it varies between runs (Raghavan, Albert and Kumara, 2007).

community_betweenness removes, one at a time, the tie with the highest edge betweenness, i.e. the tie through which most geodesics pass, as those ties connect groups rather than being inside them. The components that remain at each step give a partition, and the one with the highest modularity is returned (Girvan and Newman, 2002).

Value

These functions return the group of each node, the number of groups and the modularity of the partition.

Author(s)

Alejandro Espinosa-Rada

References

Clauset, A., Newman, M. E. J. and Moore, C. (2004). Finding community structure in very large networks. Physical Review E, 70(6), 066111. doi:10.1103/PhysRevE.70.066111

Girvan, M. and Newman, M. E. J. (2002). Community structure in social and biological networks. Proceedings of the National Academy of Sciences, 99(12), 7821–7826. doi:10.1073/pnas.122653799

Raghavan, U. N., Albert, R. and Kumara, S. (2007). Near linear time algorithm to detect community structures in large-scale networks. Physical Review E, 76(3), 036106. doi:10.1103/PhysRevE.76.036106

Examples

```
A <- matrix(c(
  0, 1, 1, 0, 0, 0,
  1, 0, 1, 0, 0, 0,
  1, 1, 0, 1, 0, 0,
  0, 0, 1, 0, 1, 1,
  0, 0, 0, 1, 0, 1,
  0, 0, 0, 1, 1, 0
), byrow = TRUE, ncol = 6)
rownames(A) <- letters[1:nrow(A)]
colnames(A) <- rownames(A)

community_greedy(A)
community_betweenness(A)
set.seed(18051889)
community_label(A)
```

---

components_id *Components*

Description

Components of a network: the groups of nodes that can reach each other.

Usage

```
components_id(A, mode = c("weak", "strong"), bipartite = FALSE)
```

Arguments

A	A square matrix, or an incidence matrix if bipartite = TRUE
mode	Whether the components are weak (default) or strong
bipartite	Whether the matrix is an incidence matrix of a two-mode network

Details

In a weak component the nodes are connected when the direction of the ties is ignored. In a strong component every node can reach every other node following the direction of the ties, so a strong component is contained in a weak one. For undirected networks both are the same.

For two-mode networks, the nodes of both sets are placed in the same network before looking for the components, so a component contains the nodes of the first mode and the ones of the second mode that they share.

Value

A vector with the component of each node, and the size of the components.

Author(s)

Alejandro Espinosa-Rada

References

Wasserman, S. and Faust, K. (1994). Social network analysis: Methods and applications. Cambridge University Press.

Examples

```
A <- matrix(c(
  0, 1, 1, 0, 0,
  1, 0, 1, 0, 0,
  1, 1, 0, 0, 0,
  0, 0, 0, 0, 1,
  0, 0, 0, 1, 0
), byrow = TRUE, ncol = 5)
rownames(A) <- letters[1:ncol(A)]
colnames(A) <- rownames(A)
components_id(A)

# In a chain of citations every node is in the same weak component,
# but each of them is its own strong component
B <- matrix(c(
  0, 1, 0, 0,
  0, 0, 1, 0,
  0, 0, 0, 1,
  0, 0, 0, 0
), byrow = TRUE, ncol = 4)
components_id(B)
components_id(B, mode = "strong")
```

---

compound_relation *Relational composition*

Description

This function returns the relational composition of the given matrices. The compound relations define the paths and the social process flows of the given matrices (Pattison, 1993). However, those whom they link may or may not be aware of them. The compound relations allow us to identify "the possibly very long and devious chains of effects propagating withing concrete social systems through links of various kinds" (Lorrain & White, 1971: 50).

Usage

```
compound_relation(l = list(), comp = 3, matrices = FALSE, equate = FALSE)
```

Arguments

l	A list of matrices.
comp	A number with the length of paths to form the compound relation.
matrices	Whether to return the resulting matrices of the compound relations.
equate	Whether to return the semigroup equations.

Value

This function provides the composition or concatenation of compound relations and the primitives of the matrices.

Author(s)

Alejandro Espinosa-Rada

References

Boorman, Scott A. and White, Harrison C. (1976) Social Structure from Multiple Networks. II. Role Structures. American Journal of Sociology. 81(6): 1384-1446.

Lorrain, Francois and White, Harrison C. (1971) Structural Equivalence of Individuals in Social Networks. Journal of Mathematical Sociology. 1: 49-80

Pattison, Philippa (1993) Algebraic Models for Social Networks. Cambridge University Press.

Examples

```
A <- matrix(c(
  0, 1, 0, 0,
  1, 0, 0, 0,
  1, 1, 0, 1,
  0, 0, 1, 0
), byrow = TRUE, ncol = 4)
rownames(A) <- letters[1:NCOL(A)]
colnames(A) <- rownames(A)

B <- matrix(c(
  0, 1, 0, 0,
  1, 0, 0, 0,
  0, 0, 0, 1,
  0, 0, 1, 0
), byrow = TRUE, ncol = 4)
rownames(B) <- letters[1:NCOL(B)]
colnames(B) <- rownames(B)

cmp <- compound_relation(list(A, B), comp = 2, matrices = TRUE, equate = TRUE)
cmp$compound_relations
cmp$compound_matrices
cmp$equated
```

concor *CONCOR*

Description

Convergence of iterated correlations (Breiger, Boorman and Arabie, 1975), a partition of the nodes into positions of structurally equivalent actors.

Usage

```
concor(A, splits = 1, max_iter = 50, tol = 1e-08)
```

Arguments

A	A square matrix, or a list of matrices of the same order for multiple relations
splits	Number of successive divisions
max_iter	Maximum number of iterated correlations
tol	Tolerance to decide that the correlations have converged

Details

The rows and the columns of the matrix describe how each node relates to the others, so they are stacked into a profile. The correlations between the profiles of every pair of nodes are computed, and the correlations of those correlations are computed again and again. The matrix converges to a matrix of ones and minus ones, which splits the nodes into two positions. The procedure is repeated within each position, so splits divisions give at most 2^{splits} positions.

Value

This function returns the position of each node and the number of positions.

Author(s)

Alejandro Espinosa-Rada

References

Breiger, R. L., Boorman, S. A. and Arabie, P. (1975). An algorithm for clustering relational data with applications to social network analysis and comparison with multidimensional scaling. Journal of Mathematical Psychology, 12(3), 328–383. doi:10.1016/00222496(75)900280

Examples

```
A <- matrix(c(
  0, 1, 1, 0, 0, 0,
  1, 0, 1, 0, 0, 0,
  1, 1, 0, 0, 0, 0,
  0, 0, 0, 0, 1, 1,
  0, 0, 0, 1, 0, 1,
  0, 0, 0, 1, 1, 0
), byrow = TRUE, ncol = 6)
rownames(A) <- letters[1:nrow(A)]
colnames(A) <- rownames(A)

concor(A, splits = 1)
```

---

core_periphery *Core-periphery structure*

Description

Core-periphery model of Borgatti and Everett (2000): a group of nodes that are connected among themselves and with the rest, and a periphery of nodes that are connected with the core but not with each other.

Usage

```
core_periphery(
  A,
  method = c("discrete", "continuous"),
  digraph = FALSE,
  rep = 50
)
```

Arguments

A	A square matrix
method	Whether to return a discrete partition (default) or continuous coreness scores
digraph	Whether the matrix is directed or undirected
rep	Number of random partitions used to start the search, besides the one given by the degree of the nodes

Details

The discrete model looks for the partition of the nodes into a core and a periphery that maximises the correlation between the observed matrix and the ideal pattern, where a tie is expected when at least one of the two nodes belongs to the core. The search starts from the nodes sorted by degree, and from rep random partitions, and then moves one node at a time while the correlation improves. As the search can end in a local optimum, the result of the random starts depends on the seed.

The `continuous` model gives each node a coreness score instead of a class. The scores maximise the correlation between the observed matrix and the products of the scores of each pair, and are given by the leading eigenvector of the matrix.

Value

This function returns the fit of the model, and the partition or the coreness scores.

Author(s)

Alejandro Espinosa-Rada

References

Borgatti, S. P. and Everett, M. G. (2000). Models of core/periphery structures. Social Networks, 21(4), 375–395. doi:10.1016/S03788733(99)000192

Examples

```
A <- matrix(c(
  0, 1, 1, 1, 1, 0,
  1, 0, 1, 1, 0, 1,
  1, 1, 0, 1, 0, 0,
  1, 1, 1, 0, 0, 0,
  1, 0, 0, 0, 0, 0,
  0, 1, 0, 0, 0, 0
), byrow = TRUE, ncol = 6)
rownames(A) <- letters[1:nrow(A)]
colnames(A) <- rownames(A)

core_periphery(A)
core_periphery(A, method = "continuous")
```

---

cug_test *Conditional uniform graph test*

Description

Compares a statistic of the observed network with its distribution in random networks that share some of its features (Anderson, Butts and Carley, 1999; Wasserman and Faust, 1994).

Usage

```
cug_test(
  A,
  FUN,
  cmode = c("edges", "size", "dyad"),
  reps = 1000,
  digraph = TRUE,
```

```
  ...
)
```

Arguments

A	A square matrix
FUN	A function that takes a matrix and returns a single number
cmode	The feature that the random networks share with the observed one: size, edges (default) or dyad
reps	Number of random networks
digraph	Whether the matrix is directed or undirected
...	Other arguments passed to FUN

Details

The random networks are drawn from a uniform distribution conditioned on:

size: only the number of nodes, so every tie is present with probability one half,

edges: the number of nodes and the number of ties,

dyad: the dyad census, i.e. the number of mutual, asymmetric and null dyads (U|MAN).

The test says whether the statistic is higher or lower than expected once those features are taken into account. Conditioning on the dyad census, for instance, removes the tendency towards reciprocity before looking at the triads.

Value

This function returns the observed statistic, the mean and the standard deviation of the distribution, and the proportion of random networks with a statistic greater or equal, and lower or equal, than the observed one.

Author(s)

Alejandro Espinosa-Rada

References

Anderson, B. S., Butts, C. and Carley, K. (1999). The interaction of size and density with graph-level indices. Social Networks, 21(3), 239–267. doi:10.1016/S03788733(99)000118

Wasserman, S. and Faust, K. (1994). Social network analysis: Methods and applications. Cambridge University Press.

Examples

```
A <- matrix(c(
  0, 1, 1, 0, 0, 0,
  1, 0, 1, 0, 0, 0,
  1, 1, 0, 1, 0, 0,
  0, 0, 1, 0, 1, 1,
```

```
  0, 0, 0, 1, 0, 1,
  0, 0, 0, 1, 1, 0
), byrow = TRUE, ncol = 6)
rownames(A) <- letters[1:nrow(A)]
colnames(A) <- rownames(A)

set.seed(18051889)
cug_test(A,
  FUN = function(x) trans_coef(x, method = "global"),
  cmode = "edges", reps = 100, digraph = FALSE
)
```

`cumulativeSumMatrices` *Cumulative sum of matrices*

Description

Cumulative sum of matrices

Usage

```
cumulativeSumMatrices(matrixList)
```

Arguments

`matrixList`	A list of matrices

Value

This function returns the cumulative sum of matrices

Author(s)

Alejandro Espinosa-Rada

Examples

```
A <- matrix(c(
  0, 1, 1,
  0, 0, 0,
  0, 1, 0
), byrow = TRUE, ncol = 3)
B <- matrix(c(
  0, 0, 1,
  0, 0, 0,
  0, 0, 0
), byrow = TRUE, ncol = 3)
C <- matrix(c(
  0, 0, 0,
  1, 0, 0,
```

```
  0, 0, 0
), byrow = TRUE, ncol = 3)
matrixList <- list(A, B, C)
cumulativeSumMatrices(matrixList)
```

---

dag *DAG validation and topological ordering for citation networks*

Description

dag_check verifies whether a directed adjacency matrix represents a directed acyclic graph (DAG) and, when it does not, removes a set of arcs that breaks every cycle. dag_sort returns the node names in topological order (sources first, sinks last).

Usage

```
dag_check(A)

dag_sort(A)
```

Arguments

A	A square, named, directed adjacency matrix.

Details

In the convention used throughout the main path functions, A[i, j] > 0 means that knowledge flows from paper i to paper j, that is, paper j cites paper i (Liu and Lu, 2012; Kuan, 2020). Sources (in-degree zero) are the papers that cite nobody in the corpus; sinks (out-degree zero) are the papers that nobody in the corpus cites. A matrix in which A[i, j] > 0 means that i cites j should be transposed first with t(A).

The order is obtained with the algorithm of Kahn (1962): the sources are placed first, their arcs are removed, and the nodes left without incoming arcs are placed next, until no node remains. When some nodes are never left without incoming arcs, they lie on a cycle.

A citation network can have cycles, for instance when two papers published at the same time cite each other, and a paper citing itself is a cycle of length one. dag_check removes the self-citations and then the arcs that point backwards in the ordering of Eades, Lin and Smyth (1993). Arcs that can be restored without closing a cycle are restored, so that no removed arc is unnecessary. The set is small but is not guaranteed to be the smallest possible, which is a hard problem. Liu, Lu and Ho (2019) discuss alternatives that keep all the citations, such as merging the papers of a cycle into one node, which should be applied before this function when the removed arcs matter.

Value

`dag_check` returns a named list:

`is_dag` Logical; `TRUE` if the original matrix was already acyclic.

`n_removed` Integer number of arcs removed, self-citations included.

`A` Adjacency matrix with the removed arcs set to zero.

`dag_sort` returns a character vector of node names in topological order.

Author(s)

Alejandro Espinosa-Rada

References

Eades, P., Lin, X. and Smyth, W.F. (1993). A fast and effective heuristic for the feedback arc set problem. Information Processing Letters. 47(6): 319-323. doi:10.1016/00200190(93)90079O.

Hummon, N.P. and Doreian, P. (1989). Connectivity in a citation network: The development of DNA theory. Social Networks. 11(1): 39-63. doi:10.1016/03788733(89)900178.

Kahn, A.B. (1962). Topological sorting of large networks. Communications of the ACM. 5(11): 558-562. doi:10.1145/368996.369025.

Liu, J.S., Lu, L.Y.Y. and Ho, M.H.C. (2019). A few notes on main path analysis. Scientometrics. 119(1): 379-391. doi:10.1007/s1119201903034x.

Examples

```
# P1 is cited by P2 and P3, which are both cited by P4
A <- matrix(c(
  0, 1, 1, 0,
  0, 0, 0, 1,
  0, 0, 0, 1,
  0, 0, 0, 0
), byrow = TRUE, nrow = 4)
rownames(A) <- c("P1", "P2", "P3", "P4")
colnames(A) <- c("P1", "P2", "P3", "P4")

dag_check(A)

# P4 is also cited by P1, which closes a cycle
A_cycle <- A
A_cycle["P4", "P1"] <- 1
dag_check(A_cycle)
dag_sort(A)
```

dir_inclusion *Neighbourhood inclusion in directed networks*

Description

Neighbourhood-inclusion preorders for directed networks (Marmulla and Brandes, 2026), which extend the vicinal preorder of the undirected case to the criteria that different families of centrality indices preserve.

Usage

```
dir_inclusion(
  A,
 type = c("radial_out", "radial_in", "hierarchical_down", "hierarchical_up", "medial"),
  strength = c("strong", "weak"),
  direction = c("dominated", "dominates")
)
```

Arguments

A	A square matrix
type	The criterion: radial_out (default), radial_in, hierarchical_down, hierarchical_up or medial
strength	Whether the neighbourhoods are open (strong, default) or closed (weak). It is ignored for the medial criterion
direction	Whether P[u, v] = 1 means that u is dominated by v (default) or that u dominates v

Details

Let $N^+(i)$ be the nodes that i sends ties to, $N^-(i)$ the ones that send ties to i, and $N[i]$ the same set including i. The strong relations use the open neighbourhoods and the weak ones the closed neighbourhoods:

radial_out: $N^+(i) \subseteq N^+(j)$, preserved by the indices that measure how far a node reaches, such as out-degree and closeness.

radial_in: $N^-(i) \subseteq N^-(j)$, the same for the ties received.

hierarchical_down: $N^+(i) \subseteq N^+(j)$ and $N^-(i) \supseteq N^-(j)$, so j sends more and receives less than i.

hierarchical_up: $N^-(i) \subseteq N^-(j)$ and $N^+(i) \supseteq N^+(j)$, so j receives more and sends less, which is the criterion of the indices of status.

medial: $N^+(i) \subseteq N^+[j]$ and $N^-(i) \subseteq N^-[j]$, with two extra conditions when i and j are adjacent, so that the advantage that the dominated node has from that tie is compensated. It is the criterion preserved by betweenness.

Value

This function returns a binary matrix `P` of the criterion asked for in `type`, where `P[u, v] = 1` when the neighbourhoods of `u` are included in those of `v`, in the `direction` asked for.

Author(s)

Alejandro Espinosa-Rada

References

Marmulla, G. and Brandes, U. (2026). Centrality in directed networks. Social Networks, 86, 23–34. doi:10.1016/j.socnet.2026.01.001

Schoch, D. and Brandes, U. (2016). Re-conceptualizing centrality in social networks. European Journal of Applied Mathematics, 27(6), 971–985. doi:10.1017/S0956792516000401

Examples

```
A <- matrix(c(
  0, 1, 1, 0, 0,
  0, 0, 1, 0, 0,
  0, 0, 0, 1, 1,
  0, 0, 0, 0, 1,
  0, 0, 0, 0, 0
), byrow = TRUE, ncol = 5)
rownames(A) <- letters[1:nrow(A)]
colnames(A) <- rownames(A)

dir_inclusion(A, type = "radial_out")
dir_inclusion(A, type = "medial")
```

---

dist_geographic *Geographical distances*

Description

This function calculate some geographical distances considering a list of places specifying their latitud and longitud. The function currently works for degree decimal or radians formats.

Usage

```
dist_geographic(
  latitude,
  longitud,
  method = c("spherical", "harvesine", "manhattan", "minkowski"),
  places = NULL,
  dd_to_radians = FALSE,
  p = NULL
)
```

Arguments

latitude	A vector with latitude
longitud	A vector with longitud
method	Whether to use the Spherical Law of Cosines spherical (default), Haversine formula harvesine, Manhattan Distance manhattan or Minkowski distance minkowski
places	A vector with the names of the places
dd_to_radians	Whether to transform degree decimal format to radians
p	Parameter p for the estimation of Minkowski distance (default = 2, which is equivalent to an Euclidian Distance)

Value

This function return a distance matrix.

Source

Adapted from Mario Pineda-Krch (Great-circle distance calculations in R)

Examples

```
set.seed(1234)
x <- cbind(latitud = rnorm(5, -90), longitud = rnorm(5, 45))
dist_geographic(x[, 1], x[, 2], method = "harvesine")
```

---

dist_sim_matrix *Structural similarities*

Description

In the literature of social network, Euclidean distance (Burt, 1976) or correlations (Wasserman and Faust, 1994) were considered as measures of structural equivalence.

Usage

```
dist_sim_matrix(
  A,
  method = c("euclidean", "hamming", "jaccard"),
  bipartite = FALSE
)
```

Arguments

A	A matrix
method	The similarities/distance currently available are either Euclidean (default), Hamming, or Jaccard.
bipartite	Whether the object is an incidence matrix

Value

This function returns a distance matrix between nodes of the same matrix.

Author(s)

Alejandro Espinosa-Rada

References

Burt, Ronald S. (1976) Positions in networks. Social Forces, 55(1): 93-122.

Wasserman, S. and Faust, K. (1994). Social network analysis: Methods and applications. Cambridge University Press.

Examples

```
A <- matrix(c(
  0, 1, 0, 0, 1,
  0, 0, 0, 1, 1,
  0, 1, 0, 0, 1,
  0, 0, 1, 1, 0,
  0, 1, 0, 0, 0
), nrow = 5, ncol = 5, byrow = TRUE)
rownames(A) <- letters[1:nrow(A)]
colnames(A) <- rownames(A)
dist_sim_matrix(A, method = "jaccard")

A <- matrix(c(
  0, 0, 3, 0, 5,
  0, 0, 2, 0, 4,
  5, 4, 0, 4, 0,
  0, 3, 0, 1, 0,
  0, 0, 0, 0, 2
), nrow = 5, ncol = 5, byrow = TRUE)
dist_sim_matrix(A, method = "euclidean")

# Several relations are compared at the same time, stacking the rows and the
# columns of every matrix
B <- matrix(c(
  0, 1, 0, 0, 1,
  1, 0, 0, 0, 1,
  0, 0, 0, 1, 0,
  0, 0, 1, 0, 0,
  1, 1, 0, 0, 0
), nrow = 5, ncol = 5, byrow = TRUE)
dist_sim_matrix(list(A, B), method = "euclidean")
```

distances *Path distances*

Description

Distances between nodes using breadth-first search (BFS) or Dijkstra's algorithm to find shortest path distances.

Usage

```
bfs_ugraph(A, from = NULL)

count_geodesics(A)

short_path(A, from = NULL, to = NULL)

wlocal_distances(A, select = c("all", "in", "out"), from, to, path = c())

wall_distances(A, select = c("all", "in", "out"))
```

Arguments

A	A symmetric matrix object
from	Node in which the path start
to	Node in which the path end
select	Whether to consider all sender and receiver ties of ego (all), only incoming ties (in), or outgoing ties (out). By default, all.
path	Path of the nodes

Value

This function returns the distances o shortest path distance between two nodes for unweighted graph (bfs_ugraph, count_geodesics and short_path respectively) and weighted graphs (wlocal_distances or wall_distances)

Author(s)

Alejandro Espinosa-Rada

References

Dijkstra, E. W. (1959). A note on two problems in connexion with graphs. Numerische Mathematik. 1: 269–271.

Examples

```
A <- matrix(c(
  0, 1, 1, 0, 0, 0,
  0, 0, 0, 1, 1, 0,
  0, 0, 0, 0, 1, 0,
  0, 0, 0, 0, 0, 0,
  0, 0, 0, 0, 0, 1,
  0, 0, 0, 0, 0, 0
), byrow = TRUE, nrow = 6)
rownames(A) <- letters[1:nrow(A)]
colnames(A) <- letters[1:ncol(A)]

bfs_ugraph(A, from = "a")


A <- matrix(c(
  0, 1, 1, 0, 0, 0,
  0, 0, 0, 1, 1, 0,
  0, 0, 0, 0, 1, 0,
  0, 0, 0, 0, 0, 0,
  0, 0, 0, 0, 0, 1,
  0, 0, 0, 0, 0, 0
), byrow = TRUE, nrow = 6)
rownames(A) <- letters[1:nrow(A)]
colnames(A) <- letters[1:ncol(A)]

count_geodesics(A)


A <- matrix(c(
  0, 1, 1, 0, 0, 0,
  0, 0, 0, 1, 1, 0,
  0, 0, 0, 0, 1, 0,
  0, 0, 0, 0, 0, 0,
  0, 0, 0, 0, 0, 1,
  0, 0, 0, 0, 0, 0
), byrow = TRUE, nrow = 6)
rownames(A) <- letters[1:nrow(A)]
colnames(A) <- letters[1:ncol(A)]

short_path(A, from = "a", to = "d")


A <- matrix(
  c(
    0, 3, 3, 10, 15, 0, 0, 0,
    1, 0, 5, 2, 7, 0, 0, 0,
    3, 5, 0, 0, 0, 0, 0, 0,
    10, 2, 0, 0, 2, 7, 12, 0,
    11, 3, 0, 3, 0, 11, 2, 0,
    0, 0, 0, 7, 11, 0, 3, 2,
    0, 0, 0, 12, 2, 3, 0, 2,
```

```
    0, 0, 0, 0, 0, 2, 2, 0
  ),
  byrow = TRUE, ncol = 8, nrow = 8
)
rownames(A) <- c("a", "b", "s", "c", "d", "e", "f", "z")
colnames(A) <- rownames(A)
wlocal_distances(A, from = "a", to = "d")


A <- matrix(
  c(
    0, 3, 3, 10, 15, 0, 0, 0,
    1, 0, 5, 2, 7, 0, 0, 0,
    3, 5, 0, 0, 0, 0, 0, 0,
    10, 2, 0, 0, 2, 7, 12, 0,
    11, 3, 0, 3, 0, 11, 2, 0,
    0, 0, 0, 7, 11, 0, 3, 2,
    0, 0, 0, 12, 2, 3, 0, 2,
    0, 0, 0, 0, 0, 2, 2, 0
  ),
  byrow = TRUE, ncol = 8, nrow = 8
)
rownames(A) <- c("a", "b", "s", "c", "d", "e", "f", "z")
colnames(A) <- rownames(A)
wall_distances(A, select = "in")
```

`dominance_layers` *Dominance layers*

Description

Layers, status and transitive reduction of a strict dominance relation.

Usage

```
dominance_layers(D, reduction = FALSE, direction = c("dominated", "dominates"))
```

Arguments

`D`	A binary dominance matrix without cycles, such as the output of `pareto_dominance()`
`reduction`	Whether to return the transitive reduction of `D`
`direction`	Whether `D[u, v] = 1` means that `u` is `dominated` by `v` (default) or that `u` `dominates` `v`

Details

The first layer contains the nodes that are not dominated by any other node (maximal elements). These nodes are removed, and the procedure is repeated until all nodes are assigned. Each layer is an antichain, i.e. the nodes within a layer do not dominate each other.

The maximal elements are of two kinds (Espinosa-Rada, 2026): `dominant` nodes dominate at least one other node, while `independent` nodes neither dominate nor are dominated. The remaining nodes are `dominated`. The net dominance is the number of nodes dominated minus the number of nodes dominating.

The transitive reduction removes the tie $u \to v$ when there is another node k such that $u \to k \to v$, which is the usual representation of a hierarchy (Hasse diagram).

Value

This function returns the layers, the layer, status and net dominance of each node and, if requested, the transitive reduction (with the same direction as `D`).

Author(s)

Alejandro Espinosa-Rada

References

Brandes, U. (2016). Network positions. Methodological Innovations, 9, 1–19. doi:10.1177/2059799116630650

Espinosa-Rada, A. (2026). Network positions within scholars and intellectual networks. Journal of Informetrics, 20, 101854. doi:10.1016/j.joi.2026.101854

Examples

```
X <- matrix(c(
  1, 1, 1, 0,
  1, 1, 0, 0,
  1, 0, 0, 0,
  0, 1, 0, 0,
  0, 0, 1, 1
), byrow = TRUE, ncol = 4)
rownames(X) <- c("a1", "a2", "a3", "a4", "a5")
colnames(X) <- c("w1", "w2", "w3", "w4")

D <- pareto_dominance(list(set_inclusion(X)))
dominance_layers(D, reduction = TRUE)
```

dominance_pairs *Dominance pairs*

Description

Comparable and incomparable pairs of a dominance relation.

Usage

```
dominance_pairs(P, direction = c("dominated", "dominates"))
```

Arguments

P	A binary dominance matrix (e.g. from neigh_inclusion())
direction	Whether P[u, v] = 1 means that u is dominated by v (default) or that u dominates v

Details

Two nodes are comparable if one is dominated by the other. Centrality indices resolve the incomparable pairs with a total ranking, which is not required by the structure of the network.

Value

This function returns the number and proportion of comparable pairs, and a table of the unordered pairs, where u < v means that u is dominated by v.

Author(s)

Alejandro Espinosa-Rada

References

Schoch, D. and Brandes, U. (2016). Re-conceptualizing centrality in social networks. European Journal of Applied Mathematics, 27(6), 971–985. doi:10.1017/S0956792516000401

Examples

```
A <- matrix(c(
  0, 1, 1, 1, 0,
  1, 0, 0, 0, 0,
  1, 0, 0, 0, 1,
  1, 0, 0, 0, 0,
  0, 0, 1, 0, 0
), byrow = TRUE, ncol = 5)
rownames(A) <- letters[1:nrow(A)]
colnames(A) <- rownames(A)

dominance_pairs(neigh_inclusion(A))
```

`dominance_ranks` *Rank intervals*

Description

The ranks that a node can take in the rankings that are consistent with a dominance relation (Schoch and Brandes, 2016).

Usage

```
dominance_ranks(P, direction = c("dominated", "dominates"))
```

Arguments

`P`	A binary dominance matrix, such as the output of `neigh_inclusion()`
`direction`	Whether `P[u, v] = 1` means that u is `dominated` by v (default) or that u `dominates` v

Details

A dominance relation only orders some pairs of nodes. Any centrality index that preserves it gives a complete ranking, but different indices give different rankings. The interval of a node contains every rank it can take in such a ranking: it cannot be ranked below the nodes it dominates, nor above the nodes that dominate it. A node with a wide interval is one whose position depends on the index that is chosen, and a node with an interval of a single value has the same rank under every index that preserves the relation.

The rank one is the lowest.

Value

This function returns the minimum and the maximum rank of every node, and the width of the interval.

Author(s)

Alejandro Espinosa-Rada

References

Schoch, D. and Brandes, U. (2016). Re-conceptualizing centrality in social networks. European Journal of Applied Mathematics, 27(6), 971–985. doi:10.1017/S0956792516000401

Examples

```
A <- matrix(c(
  0, 1, 1, 1, 0,
  1, 0, 0, 0, 0,
  1, 0, 0, 0, 1,
  1, 0, 0, 0, 0,
  0, 0, 1, 0, 0
), byrow = TRUE, ncol = 5)
rownames(A) <- letters[1:nrow(A)]
colnames(A) <- rownames(A)

dominance_ranks(neigh_inclusion(A))
```

---

dyad_triad_table *Forbidden triad table*

Description

This function explores dyads and triads (Simmel, 1950), building from the 'forbidden triad' (Granovetter, 1973). First, the minimum structure is an isolated node, then dyads. Afterwards, different combinations of 'forbidden triads' are explored.

Usage

```
dyad_triad_table(A, adjacency_list = FALSE, min = NULL, max = NULL)
```

Arguments

A	A symmetric matrix object.
adjacency_list	Whether to return the adjacency list of the triads per node.
min	Numeric constant, lower limit on the number of forbidden triads (201) of which a node is the centre. NULL means no limit.
max	Numeric constant, upper limit on the number of forbidden triads (201) of which a node is the centre. NULL means no limit.

Details

For each node, every pair of its neighbours forms a triad with the node at its centre. The triad is a forbidden triad (type 201) when the two neighbours are not tied, which Granovetter (1973) argued is unlikely when both ties are strong, and it is closed (type 300) when they are. A node with a single neighbour is listed with its dyad (type 102) and an isolated node alone (type 003). The underlying graph of the network is used.

The same triad receives the same number in triad for every node that lists it: a closed triad is listed by its three nodes, and a forbidden triad only by its centre.

Value

This function returns a data frame with the triads of each node: the `node`, the number of the `triad`, its `members` and its `type`.

If `adjacency_list = TRUE` it also return the adjacency list of the triads per node.

Author(s)

Alejandro Espinosa-Rada

References

Granovetter, M.S. (1973). The Strength of Weak Ties. American Journal of Sociology. 78 (6): 1360–80. doi:10.1086/225469.

Simmel, G. (1950). Individual and Society. In K. H. Wolff (Ed.), The Sociology of George Simmel. New York: Free Press.

Wasserman, S. and Faust, K. (1994). Social network analysis: Methods and applications. Cambridge University Press.

Examples

```
A <- matrix(c(
  0, 1, 1, 1, 0,
  1, 0, 1, 0, 0,
  1, 1, 0, 0, 0,
  1, 0, 0, 0, 1,
  0, 0, 0, 1, 0
), byrow = TRUE, ncol = 5)
rownames(A) <- letters[1:nrow(A)]
colnames(A) <- letters[1:ncol(A)]

dyad_triad_table(A)

# Nodes at the centre of at least two forbidden triads
dyad_triad_table(A, adjacency_list = TRUE, min = 2)
```

---

dyadic_census *Dyad census*

Description

Dyad census

Usage

```
dyadic_census(G, directed = TRUE, loops = FALSE)
```

Arguments

G	A symmetric matrix object.
directed	Whether the matrix is directed or not
loops	Whether to expect nonzero elements in the diagonal of the matrix

Value

This function return the counts of the dyad census.

Author(s)

Alejandro Espinosa-Rada

References

Wasserman, S. and Faust, K. (1994). Social network analysis: Methods and applications. Cambridge University Press.

Examples

```
data(krackhardt_friends)
dyadic_census(krackhardt_friends)

data(FIFAin)
dyadic_census(FIFAin[[1]], directed = FALSE)
```

---

eb_constraint *Constraint*

Description

Everett and Borgatti specification of the constraint measure for binary, directed and valued matrices

Usage

```
eb_constraint(A, ego = NULL, digraph = FALSE, weighted = FALSE)
```

Arguments

A	A matrix object
ego	Name of ego in the matrix
digraph	Whether the matrix is directed or undirected
weighted	Whether the matrix is weighted or not

Details

The constraint of Burt (1992) is computed in the ego network, from the proportion p_{ij} of the ties of each node i that go to j, and is split in the three terms of Everett and Borgatti (2020: Eq. 2):

$$\sum_j p_{ij}^2 + 2\sum_j p_{ij} \sum_q p_{iq}p_{qj} + \sum_j \left(\sum_q p_{iq}p_{qj} \right)^2$$

For binary undirected networks, the first term is one over the number of alters N.

Burt (1992) uses the ties in both directions, $p_{ij} \propto a_{ij} + a_{ji}$, so the constraint of a directed network is that of the undirected valued network $A + A^T$, in which a reciprocated tie counts twice and an unreciprocated tie once (Everett and Borgatti, 2020: 53).

The normalization is $(c-1/N)/(c_{max}-1/N)$, where $1/N$ is the minimum and c_{max} the maximum constraint of an ego with N alters. The maximum is reached in a complete ego network or in a shadow ego network, in which one alter is tied to all the others and there are no other ties among alters (Everett and Borgatti, 2020: Eq. 4, 6, 7, 8 and 9). For valued networks the maximum depends on the smallest (m) and the largest (M) value of the ties in the ego network, and binary networks are the case $m = M = 1$. The maximum is a conjecture of Everett and Borgatti, checked by enumerating ego networks. An ego with a single alter has a normalized constraint of one.

Value

This function returns term 1, 2 and 3, the normalization and the maximum value of the specification of Everett and Borgatti (2020), and the constraint of Burt (1992).

Author(s)

Alejandro Espinosa-Rada

References

Burt, R.S., 1992. Structural Holes: the Social Structure of Competition. Harvard University Press, Cambridge.

Everett, M.G. and Borgatti, S., 2020. Unpacking Burt's constraint measure. Social Networks 62, pp. 50-57. doi:10.1016/j.socnet.2020.02.001

Examples

```
A <- matrix(c(
  0, 1, 1, 0, 0, 1,
  1, 0, 1, 0, 0, 1,
  1, 1, 0, 0, 0, 1,
  0, 0, 0, 0, 1, 1,
  0, 0, 0, 1, 0, 1,
  1, 1, 1, 1, 1, 0
), ncol = 6, byrow = TRUE)

rownames(A) <- letters[1:nrow(A)]
colnames(A) <- letters[1:ncol(A)]
eb_constraint(A, ego = "f")
```

```
# Directed network: f -> a is not reciprocated
D <- A
D["a", "f"] <- 0
eb_constraint(D, ego = "f", digraph = TRUE)

# Valued network
W <- A
W["f", "a"] <- W["a", "f"] <- 3
eb_constraint(W, ego = "f", weighted = TRUE)
```

`edgelist_to_matrix` *Transform an edgelist to a matrix*

Description

Transform an edgelist to a matrix

Usage

```
edgelist_to_matrix(
  E,
  digraph = TRUE,
  label = NULL,
  label2 = NULL,
  bipartite = FALSE,
  valued = FALSE,
  loops = FALSE,
  rule = c("weak", "strong")
)
```

Arguments

`E`	An edge list
`digraph`	Whether the matrix is directed or not
`label`	A vector with the names of the nodes, which gives their order in the matrix and adds the nodes without ties
`label2`	A vector with the names of the nodes of the second mode, with the same role as `label`, when `bipartite = TRUE`
`bipartite`	Whether the matrix is bipartite
`valued`	Whether the third column of the edgelist has the value of the tie
`loops`	Whether to keep the ties of a node with itself
`rule`	For `digraph = FALSE`, whether an undirected tie is kept when it is listed in either order (`weak`, default) or only when it is listed in both orders (`strong`), as in `sna::symmetrize`

Details

With `digraph = FALSE` each undirected tie is placed in both cells of the matrix, so the number of ties is the number of cells of one triangle (or half the sum of a binary matrix). A tie listed in both orders is a single tie.

The rows and the columns follow the order of `label`. The nodes that are not in `label`, or all the nodes when it is not given, are added in alphabetical order.

Value

This function transform the edgelist into a matrix

Author(s)

Alejandro Espinosa-Rada

Examples

```
A <- matrix(c(
  0, 1, 1, 0, 0, 0, 0, 1, 0,
  1, 0, 1, 0, 0, 0, 0, 0, 0,
  1, 1, 0, 0, 0, 0, 0, 0, 0,
  0, 0, 0, 0, 1, 1, 0, 0, 0,
  0, 0, 0, 1, 0, 0, 0, 0, 0,
  0, 0, 0, 1, 0, 0, 1, 1, 0,
  0, 0, 0, 0, 0, 1, 0, 1, 0,
  1, 0, 0, 0, 0, 1, 1, 0, 0,
  0, 0, 0, 0, 0, 0, 0, 0, 0
), byrow = TRUE, ncol = 9)
rownames(A) <- letters[1:nrow(A)]
colnames(A) <- rownames(A)
E <- matrix_to_edgelist(A)
edgelist_to_matrix(E, label = c("i"), digraph = FALSE)

# With a third column, the ties keep their value
V <- rbind(
  c("a", "b", 3),
  c("b", "c", 1),
  c("c", "a", 7)
)
edgelist_to_matrix(V, valued = TRUE)
```

---

ego_net *Ego network*

Description

Submatrix of ego's neighbourhoods

Usage

```
ego_net(
  A,
  ego = NULL,
  bipartite = FALSE,
  addEgo = FALSE,
  select = c("all", "in", "out")
)
```

Arguments

A	A symmetric matrix object
ego	Name of ego in the matrix
bipartite	Whether the matrix is a two-mode network
addEgo	Whether to retain ego in the submatrix or not
select	Whether to consider all sender and receiver ties of ego (all), only incoming ties (in), or outgoing ties (out). By default, all.

Value

This function returns the submatrix of the alters of ego, with ego in the last row and column when addEgo = TRUE. An isolate gives an empty matrix (or a 1 x 1 matrix with ego).

Author(s)

Alejandro Espinosa-Rada

References

Burt, R.S., 1992. Structural Holes: the Social Structure of Competition. Harvard University Press, Cambridge.

Borgatti, S., 1997. Unpacking Burt's redundancy measure. Connections, 20(1): 35-38.

Examples

```
A <- matrix(c(
  0, 1, 0, 0, 1, 1, 1,
  1, 0, 0, 1, 0, 0, 1,
  0, 0, 0, 0, 0, 0, 1,
  0, 1, 0, 0, 0, 0, 1,
  1, 0, 0, 0, 0, 0, 1,
  1, 0, 0, 0, 0, 0, 1,
  1, 1, 1, 1, 1, 1, 0
), ncol = 7, byrow = TRUE)
rownames(A) <- letters[1:nrow(A)]
colnames(A) <- letters[1:ncol(A)]
ego_net(A, ego = "g")
```

ei_index *Krackhardt and Stern's E-I index*

Description

This index was proposed by Krackhardt and Stern (1988) to distinguish between the relative prevalence of between and within-group ties. This measure can be interpreted as homophily at the network level.

Usage

```
ei_index(A, mixed = TRUE, att = NULL)
```

Arguments

A	A symmetric matrix object, or a mixing matrix if no attribute is given
mixed	Whether the matrix provided is already a mixing matrix. It is only used when no attribute is given
att	Categorical attribute of the nodes. When it is given, the mixing matrix is computed from A

Value

Numerical value of the E-I index.

Examples

```
set.seed(18051889)
n <- 100
A <- matrix(c(rbinom(n, 1, 0.5)),
  ncol = sqrt(n), nrow = sqrt(n), byrow = TRUE
)
rownames(A) <- letters[1:nrow(A)]
colnames(A) <- letters[1:ncol(A)]

att <- rbinom(sqrt(n), 3, 0.5)
ei_index(A, att = att)

# All the ties are within the groups, so the index is -1
B <- matrix(0, 6, 6)
B[1:3, 1:3] <- 1
B[4:6, 4:6] <- 1
diag(B) <- 0
rownames(B) <- letters[1:6]
colnames(B) <- rownames(B)
ei_index(B, att = c(1, 1, 1, 2, 2, 2))
```

`eigenvector_centrality` *Eigenvector centrality*

Description

Eigenvector centrality of Bonacich (1972), the leading eigenvector of the matrix.

Usage

```
eigenvector_centrality(
  A,
  digraph = TRUE,
  type = c("in", "out"),
  weighted = FALSE,
  scale = c("max", "unit"),
  signed = FALSE
)
```

Arguments

`A`	A square matrix
`digraph`	Whether the matrix is directed or undirected
`type`	Whether to use the `in` (default) or `out` ties for directed networks
`weighted`	Whether the matrix is weighted
`scale`	Whether the vector is scaled with a maximum of one (`max`, default) or has unit length (`unit`)
`signed`	Whether the matrix has negative ties (Bonacich and Lloyd, 2004). The scores can then be negative, and the eigenvector is the one of the eigenvalue with the largest absolute value

Details

A node is central when it is connected to other central nodes. For directed networks, the `in` option gives centrality to the nodes that receive ties from central nodes, and `out` to the nodes that send ties to central nodes.

Value

This function returns the eigenvector centrality of the nodes and the leading eigenvalue.

Author(s)

Alejandro Espinosa-Rada

References

Bonacich, P. (1972). Factoring and weighting approaches to status scores and clique identification. Journal of Mathematical Sociology, 2(1), 113–120. doi:10.1080/0022250X.1972.9989806

Bonacich, P. (1987). Power and centrality: A family of measures. American Journal of Sociology, 92(5), 1170–1182. doi:10.1086/228631

Bonacich, P. and Lloyd, P. (2004). Calculating status with negative relations. Social Networks, 26(4), 331–338. doi:10.1016/j.socnet.2004.08.007

Examples

```
A <- matrix(c(
  0, 1, 1, 1, 0,
  1, 0, 0, 0, 0,
  1, 0, 0, 0, 1,
  1, 0, 0, 0, 0,
  0, 0, 1, 0, 0
), byrow = TRUE, ncol = 5)
rownames(A) <- letters[1:nrow(A)]
colnames(A) <- rownames(A)

eigenvector_centrality(A, digraph = FALSE)
eigenvector_centrality(A, digraph = FALSE, scale = "unit")

# With negative ties the status of a node can be negative
S <- matrix(c(
  0, 1, 1, -1,
  1, 0, 1, -1,
  1, 1, 0, -1,
  -1, -1, -1, 0
), byrow = TRUE, ncol = 4)
rownames(S) <- letters[1:nrow(S)]
colnames(S) <- rownames(S)

eigenvector_centrality(S, digraph = FALSE, signed = TRUE)
```

---

expand_matrix *Expand Matrix*

Description

Expand Matrix

Usage

```
expand_matrix(A, label = NULL, loops = FALSE, normalize = FALSE)
```

Arguments

A	A square matrix
label	Duplicated labels to expand the matrix
loops	Whether the loops are retained or not
normalize	Whether to normalize the matrix considering the fractional counting per group

Value

Return an expanded matrix

Author(s)

Alejandro Espinosa-Rada

Examples

```
A <- matrix(c(
  0, 1, 1,
  0, 0, 1,
  1, 0, 0
), byrow = TRUE, ncol = 3, nrow = 3)
rownames(A) <- letters[1:NROW(A)]
colnames(A) <- rownames(A)
label <- sort(rep(rownames(A), 2))
expand_matrix(A, label, loops = FALSE, normalize = TRUE)
```

---

extract_component *Extract components*

Description

This function extract the matrix of different components

Usage

```
extract_component(A, maximum = TRUE, position = NULL)
```

Arguments

A	A matrix
maximum	Whether to extract the maximum component
position	The position of the size of the component, from the largest (1). Used when maximum = FALSE

Value

The matrix of the component, or a list with the matrices of the components when several have the same size

Author(s)

Alejandro Espinosa-Rada

References

Wasserman, S. and Faust, K. (1994). Social network analysis: Methods and applications. Cambridge University Press.

Examples

```
A <- FIFAex$Matrix
rownames(A) <- FIFAex$label
colnames(A) <- rownames(A)
extract_component(A, maximum = TRUE)
extract_component(A, maximum = FALSE, position = 2)
```

FIFAego *Ego FIFA*

Description

Multilevel Network of the regulatory transnational regime of the International Federation of Association Football (FIFA)

Usage

```
data(FIFAego)
```

Format

A list of a 48 x 48 symmetric matrix of the ego network of FIFA as a different entity, and a string vector with the label of the actors

Source

Espinosa, Alejandro & Ortiz, Francisca (2016). "Jurisdictional autonomy in the regulatory transnational regime of FIFA". REDES- Revista Hispana para el Analisis de Redes Sociales, 27(1), 100-112. (Original title in Spanish: "Autonomia jurisdiccional en el reegimen regulatorio transnacional de la FIFA") doi:10.5565/rev/redes.595

FIFAex *Outside FIFA*

Description

Multilevel Network of the regulatory transnational regime of the International Federation of Association Football (FIFA)

Usage

```
data(FIFAex)
```

Format

A list of a 7 x 7 symmetric matrix of non-FIFA organizations, and a string vector with the label of the actors

Source

Espinosa, Alejandro & Ortiz, Francisca (2016). "Jurisdictional autonomy in the regulatory transnational regime of FIFA". REDES- Revista Hispana para el Analisis de Redes Sociales, 27(1), 100-112. (Original title in Spanish: "Autonomia jurisdiccional en el reegimen regulatorio transnacional de la FIFA") doi:10.5565/rev/redes.595

FIFAin *Inside FIFA*

Description

Multilevel Network of the regulatory transnational regime of the International Federation of Association Football (FIFA)

Usage

```
data(FIFAin)
```

Format

A list of a 41 x 41 symmetric matrix of roles and organizations inside FIFA, and a string vector with the label of the actors

Source

Espinosa, Alejandro & Ortiz, Francisca (2016). "Jurisdictional autonomy in the regulatory transnational regime of FIFA". REDES- Revista Hispana para el Analisis de Redes Sociales, 27(1), 100-112. (Original title in Spanish: "Autonomia jurisdiccional en el reegimen regulatorio transnacional de la FIFA") doi:10.5565/rev/redes.595

`fractional_approach` *Fractional approach*

Description

Citation, co-citation and bibliographic coupling networks with full or fractional counting (Batagelj, 2020).

Usage

```
fractional_approach(
  A1,
  A2 = NULL,
  approach = c("citation", "cocitation", "bcoupling"),
  fractional = TRUE,
  symmetric = c("none", "average", "minimum", "maximum", "geometric", "harmonic",
    "jaccard")
)
```

Arguments

`A1`	A citation network between works, where `A1[p, q] = 1` if the work p cites the work q
`A2`	For `approach = "citation"`, the authorship matrix, with the works in rows and the authors in columns
`approach`	Character string, "citation", "cocitation" and "bcoupling"
`fractional`	Whether to use fractional counting (default) or full counting
`symmetric`	For fractional bibliographic coupling, the symmetric similarity: `none` (default, the proportion of shared references), `average`, `minimum`, `maximum`, `geometric`, `harmonic` or `jaccard`

Details

`A1` is a citation network Ci between works, where `A1[p, q] = 1` when the work p cites the work q. With fractional counting each work has a total weight of one, which is divided equally among the works it cites: $Cin = D \cdot Ci$, where D is the diagonal matrix of one over the number of references of each work (one when it has none). Fractional counting prevents the works with many references, such as reviews, from dominating the result (Batagelj, 2020).

`approach = "cocitation"`: $Ci^T Ci$, the number of works that cite both works, or with fractional counting $Cin^T Cin$, in which each citing work contributes a total of one.

`approach = "bcoupling"`: $Ci \cdot Ci^T$, the number of works cited by both works. Fractional counting cannot be applied in the same way to bibliographic coupling (Batagelj, 2020: 631), so it gives $biC = Cin \cdot Ci^T$, the proportion of the references of p that it shares with q, which is not symmetric. `symmetric` turns it into a symmetric similarity: the `average`, the `minimum`, the `maximum`,

the `geometric` mean (the cosine of Salton), the `harmonic` mean, or the `jaccard` index, the shared references divided by the references of either work.

`approach = "citation"`: the citations between the authors of the works, given the authorship matrix WA in `A2` (works in rows, authors in columns): $WA^T \cdot Ci \cdot WA$, the number of times the works of an author cite the works of another. With fractional counting each work is divided equally among its authors, $WAn^T \cdot Ci \cdot WAn$, so that the total of the network is the number of citations.

Which count should be conserved guides the choice between the two protocols (Prathap and Mukherjee, 2020): full counting conserves the number of paths between the nodes, and fractional counting the number of nodes (works).

Value

Return the citation network between authors, the co-citation network or the bibliographic coupling network.

Author(s)

Alejandro Espinosa-Rada

References

Batagelj, V. (2020). On fractional approach to analysis of linked networks. Scientometrics, 123(2), 621-633. doi:10.1007/s1119202003383y

Batagelj, V. (2022). Analysis of the Southern women network using fractional approach. Social Networks, 68, 229-236 doi:10.1016/j.socnet.2021.08.001

Batagelj, V., & Cerinšek, M. (2013). On bibliographic networks. Scientometrics, 96(3), 845–864. doi:10.1007/s1119201209401

Prathap, G., & Mukherjee, S. (2020). Letter to the Editor: Comments on the paper of Batagelj—on fractional approach to analysis of linked networks. Scientometrics, 124(3), 2717–2722. doi:10.1007/s11192020035412

Examples

```
# Five works: w1 cites w2 and w3, w4 cites w2, w3 and w5, w5 cites w3
Ci <- matrix(c(
  0, 1, 1, 0, 0,
  0, 0, 0, 0, 0,
  0, 0, 0, 0, 0,
  0, 1, 1, 0, 1,
  0, 0, 1, 0, 0
), byrow = TRUE, ncol = 5)
rownames(Ci) <- colnames(Ci) <- paste0("w", 1:5)

# Authors of the works
WA <- matrix(c(
  1, 1, 0,
  0, 1, 0,
  0, 0, 1,
  1, 0, 0,
```

```
  0, 0, 1
), byrow = TRUE, ncol = 3)
rownames(WA) <- rownames(Ci)
colnames(WA) <- c("a1", "a2", "a3")

fractional_approach(Ci, WA, approach = "citation")
fractional_approach(Ci, approach = "cocitation")
fractional_approach(Ci, approach = "bcoupling", symmetric = "geometric")
```

---

gen_degree *Generalized degree*

Description

Generalized degree centrality for one-mode and bipartite networks

Usage

```
gen_degree(
  A,
  weighted = FALSE,
  type = "out",
  normalized = FALSE,
  loops = TRUE,
  digraph = TRUE,
  alpha = 0.5,
  bipartite = FALSE
)
```

Arguments

A	A matrix object
weighted	Whether the matrix is weighted or not
type	Character string, "out" (outdegree), "in" (indegree) and "all" (degree)
normalized	Whether normalize the measure for the one-mode network (Freeman, 1978) or a bipartite network (Borgatti and Everett, 1997)
loops	Whether the diagonal of the matrix is considered or not
digraph	Whether the matrix is directed or undirected
alpha	Sets the alpha parameter in the generalised measures from Opsahl et al. (2010)
bipartite	Whether the matrix is bipartite or not.

Value

This function returns term 1, 2 and 3, the normalization and the maximum value of the specification of Everett and Borgatti (2020), and the constraint of Burt (1992)

Author(s)

Alejandro Espinosa-Rada

References

Borgatti, S. P., and Everett, M. G. (1997). Network analysis of 2-mode data. Social Networks, 19(3), 243–269.

Freeman, L. C. (1978). Centrality in social networks conceptual clarification. Social Networks, 1(3), 215–239.

Opsahl, T., Agneessens, F., and Skvoretz, J. (2010). Node centrality in weighted networks: Generalizing degree and shortest paths. Social Networks, 32(3), 245–251.

Examples

```
A3 <- matrix(c(
  0, 4, 4, 0, 0, 0,
  4, 0, 2, 1, 1, 0,
  4, 2, 0, 0, 0, 0,
  0, 1, 0, 0, 0, 0,
  0, 1, 0, 0, 0, 7,
  0, 0, 0, 0, 7, 0
), byrow = TRUE, ncol = 6)

gen_degree(A3, digraph = FALSE, weighted = TRUE)
```

---

gen_density *Generalized density*

Description

Generalized density

Usage

```
gen_density(
  A,
  directed = TRUE,
  bipartite = FALSE,
  loops = FALSE,
  weighted = FALSE,
  multilayer = FALSE
)
```

Arguments

A	A symmetric or incidence matrix object
directed	Whether the matrix is directed
bipartite	Whether the matrix is bipartite
loops	Whether to consider the loops
weighted	Whether the matrix is weighted
multilayer	Whether the matrix is multilayer (i.e., multiplex and/or multilevel)

Details

The density is the number of ties divided by the number of possible ties: $n(n-1)$ for a directed network, $n(n-1)/2$ for an undirected one, and nm for a two-mode network of n and m nodes. With loops = TRUE the diagonal is counted among the possible ties of a one-mode network. With directed = FALSE, a tie in either direction is an edge of the underlying graph.

In a list of matrices (multilayer = TRUE), the rectangular matrices are taken as two-mode networks and the square ones as one-mode networks, so a square incidence matrix should be given on its own with bipartite = TRUE.

Value

This function returns the density of the matrix(es)

Author(s)

Alejandro Espinosa-Rada

References

Wasserman, S., and Faust, K. (1994). Social Network Analysis: Methods and Applications. Cambridge: Cambridge University Press.

Examples

```
# A bipartite matrix
B <- matrix(c(
  1, 1, 0,
  0, 0, 1,
  0, 1, 1,
  0, 0, 1
), byrow = TRUE, ncol = 3)
gen_density(B, bipartite = TRUE)

# A multilevel network
A1 <- matrix(c(
  0, 1, 0, 0, 1,
  1, 0, 0, 1, 1,
  0, 0, 0, 1, 1,
  0, 1, 1, 0, 1,
```

```
  1, 1, 1, 1, 0
), byrow = TRUE, ncol = 5)

B1 <- matrix(c(
  1, 0, 0,
  1, 1, 0,
  0, 1, 0,
  0, 1, 0,
  0, 1, 1
), byrow = TRUE, ncol = 3)

A2 <- matrix(c(
  0, 1, 1,
  1, 0, 0,
  1, 0, 0
), byrow = TRUE, nrow = 3)

B2 <- matrix(c(
  1, 1, 0, 0,
  0, 0, 1, 0,
  0, 0, 1, 1
), byrow = TRUE, ncol = 4)

A3 <- matrix(c(
  0, 1, 3, 1,
  1, 0, 0, 0,
  3, 0, 0, 5,
  1, 0, 5, 0
), byrow = TRUE, ncol = 4)

matrices <- list(A1, B1, A2, B2, A3)
gen_density(matrices, multilayer = TRUE)

# A multiplex network
A <- matrix(c(
  0, 1, 3, 6, 4,
  2, 0, 4, 5, 2,
  4, 1, 0, 6, 1,
  5, 6, 3, 0, 6,
  1, 1, 2, 3, 0
), byrow = TRUE, ncol = 5)
gen_density(A, multilayer = TRUE)
```

`geodesics` *Geodesic distances*

Description

Matrix of geodesic distances and a summary of the distances of the network.

Usage

```
geo_distances(
  A,
  digraph = TRUE,
  type = c("out", "in", "all"),
  weighted = FALSE,
  alpha = 1
)

geo_summary(A, digraph = TRUE, weighted = FALSE, alpha = 1)
```

Arguments

A	A square matrix
digraph	Whether the matrix is directed or undirected
type	Whether to use the out (default), in or all distances
weighted	Whether the matrix is weighted
alpha	The tuning parameter of Opsahl et al. (2010) to transform weights into lengths

Details

geo_distances returns the length of the shortest path between every pair of nodes, computed with the Floyd-Warshall algorithm in matrix form. The distance is infinite when there is no path. For valued matrices, the weights are treated as strengths and transformed into lengths as $1/w^{\alpha}$ (Opsahl et al., 2010).

geo_summary returns the diameter (the longest geodesic distance), the average distance and the proportion of ordered pairs that can reach each other. When the network is disconnected, both the diameter and the average distance only consider the pairs that are connected by a path.

Value

geo_distances returns a matrix of distances, and geo_summary the diameter, the average distance and the proportion of reachable pairs.

Author(s)

Alejandro Espinosa-Rada

References

Opsahl, T., Agneessens, F., and Skvoretz, J. (2010). Node centrality in weighted networks: Generalizing degree and shortest paths. Social Networks, 32(3), 245–251. doi:10.1016/j.socnet.2010.03.006

Wasserman, S. and Faust, K. (1994). Social network analysis: Methods and applications. Cambridge University Press.

Examples

```
A <- matrix(c(
  0, 1, 1, 0, 0, 0,
  0, 0, 0, 1, 1, 0,
  0, 0, 0, 0, 1, 0,
  0, 0, 0, 0, 0, 0,
  0, 0, 0, 0, 0, 1,
  0, 0, 0, 0, 0, 0
), byrow = TRUE, nrow = 6)
rownames(A) <- letters[1:nrow(A)]
colnames(A) <- letters[1:ncol(A)]

geo_distances(A)
geo_summary(A)
```

---

heterogeneity *Blau's and IQV index*

Description

This index was used by Blau (1977) to distinguish between the relative prevalence of between and within-group ties. This measure can be interpreted as heterogeneity at the network level.

Usage

```
heterogeneity(att, normalized = FALSE)
```

Arguments

`att`	Categorical attribute of the nodes
`normalized`	Whether to return IQV index

Value

Numerical value of the Blau index.

If `normalized = TRUE`, then the function also return IQV index.

References

Agresti, A. and Agresti, B. (1978). Statistical Analysis of Qualitative Variation. Sociological Methodology, 9, 204-237. doi:10.2307/270810

Blau, P. M. (1977). Inequality and heterogeneity. New York: Free Press.

Examples

```
a <- rep(1:10, 10)
heterogeneity(a, normalized = TRUE)

a <- rep(1:2, 10)
heterogeneity(a, normalized = TRUE)
```

---

hyperevent_dominance *Hyper-event dominance*

Description

Dominance among authors based on the hyper-event chain Author -> Citing paper -> Cited paper -> Author (Espinosa-Rada, 2026).

Usage

```
hyperevent_dominance(
  X,
  W,
  Xb = X,
  tau = 2,
  dimensions = c("authored", "cited_papers", "cited_authors"),
  closed = c(FALSE, TRUE, TRUE),
  strict = c("asymmetric", "proper"),
  closure_papers = c("citing", "authored"),
  max_authors = NULL,
  team_size = NULL,
  direction = c("dominated", "dominates")
)
```

Arguments

`X`	An incidence matrix of authors (rows) and the citing papers they authored (columns)
`W`	A square citation matrix where `W[p, q] = 1` if paper `p` cites paper `q`
`Xb`	An incidence matrix of authors and the cited papers they authored. By default, the same as `X`
`tau`	Minimum number of dimensions in which an author should be weakly dominated
`dimensions`	The dimensions to be considered: `authored`, `cited_papers` and/or `cited_authors`
`closed`	A logical vector with whether the neighbourhoods of `authored`, `cited_papers` and `cited_authors` are closed
`strict`	Whether the strict dominance is `asymmetric` (default) or a `proper` inclusion
`closure_papers`	Whether the closed neighbourhood of cited papers adds the `citing` papers of the author (default) or every paper `authored`

max_authors	If not NULL, the citing papers with more authors than this number are excluded
team_size	Number of authors of each paper, in the same order as the columns of X. By default, the column sums of X
direction	Whether D[u, v] = 1 means that u is dominated by v (default) or that u dominates v, as in the figures of Espinosa-Rada (2026)

Details

Each author has three neighbourhoods built from the hyper-events in which the author wrote the citing paper:

(authored) productive participation: the citing papers authored, X,

(cited_papers) citation reach: the cited papers, $X \circ W$. Its closed version adds the papers authored,

(cited_authors) recognition: the cited authors, $X \circ W \circ X_b^T$. Its closed version adds the author itself.

Author a_i dominates a_j when the neighbourhood of a_j is included in the (closed) neighbourhood of a_i in at least tau dimensions, and the dominance is strict in at least one dimension (see pareto_dominance()). Authors without hyper-events are excluded.

The defaults reproduce the analysis of Espinosa-Rada (2026): the three dimensions, open neighbourhoods for the authored papers and closed neighbourhoods for the cited papers and authors, and asymmetric strict dominance. The alternatives are:

strict = "proper" follows the formal definition of strict dominance as proper inclusion, $N_k(a_j) \subsetneq N_k^*(a_i)$. As closed neighbourhoods add elements to a_i, proper inclusion is less demanding than the asymmetric criterion, where a_i should not be included in a_j. With proper inclusion, two authors might dominate each other, so the relation is not always a partial order. The same might happen with the asymmetric criterion when tau = 1, if each author dominates the other in a different dimension.

closure_papers sets which papers of a_i are added to close the neighbourhood of cited papers. With "citing" (default, as in the analysis scripts of the article) they are the citing papers of a_i with hyper-events. With "authored" they are every paper authored by a_i, in X or in Xb, as in the text of Section 3.5, so that a paper of a_i cited by a_j is in the closed neighbourhood of a_i even when it does not cite other papers of the corpus.

max_authors excludes the citing papers of large teams before computing the neighbourhoods, as a robustness check for consortium papers (in the article, papers with more than 20 authors). As the matrices might only contain some of the authors of each paper (e.g. a bounded population), the number of authors can be given in team_size.

Value

This function returns a binary dominance matrix D of the authors.

Author(s)

Alejandro Espinosa-Rada

References

Espinosa-Rada, A. (2026). Network positions within scholars and intellectual networks. Journal of Informetrics, 20, 101854. doi:10.1016/j.joi.2026.101854

Schoch, D. and Brandes, U. (2016). Re-conceptualizing centrality in social networks. European Journal of Applied Mathematics, 27(6), 971–985. doi:10.1017/S0956792516000401

Examples

```
X <- matrix(c(
  1, 1, 1, 0,
  1, 1, 0, 0,
  1, 0, 0, 0,
  0, 1, 0, 0,
  0, 0, 1, 1
), byrow = TRUE, ncol = 4)
rownames(X) <- c("a1", "a2", "a3", "a4", "a5")
colnames(X) <- c("w1", "w2", "w3", "w4")

W <- matrix(c(
  0, 1, 1, 0,
  0, 0, 1, 0,
  0, 0, 0, 1,
  0, 0, 0, 0
), byrow = TRUE, ncol = 4)
rownames(W) <- colnames(X)
colnames(W) <- colnames(X)

hyperevent_dominance(X, W, tau = 2)
hyperevent_dominance(X, W, tau = 2, strict = "proper")
hyperevent_dominance(X, W, tau = 1, dimensions = c("authored", "cited_authors"))
```

---

hypergraph *Hypergraphs*

Description

Hypergraph consist of a set of objects and a collection of subsets of objects, in which each object belongs to at least one subset, and no subset is empty (Berge, 1989)

Usage

```
hypergraph(A, dual = TRUE, both = TRUE)
```

Arguments

A	An incidence matrix.
dual	Whether to return the dual hypergraph (which rever the role of the pointes and the edges)
both	Whether to return the hypergraph and the dual hypergraph

Value

This function returns an adjacent list of the subsets of entities in the hypergraph.

Author(s)

Alejandro Espinosa-Rada

References

Berge, C. (1973). Graphs and hypergraphs.Amsterdam: North-Holland.

Berge, C. (1989). Hypergraphs: Combinatorics of finite sets. Amsterdam: North-Holland.

Wasserman, S. and Faust, K. (1994). Social network analysis: Methods and applications. Cambridge University Press.

Examples

```
A <- matrix(c(
  1, 0, 1,
  0, 1, 0,
  0, 1, 1,
  0, 0, 1,
  1, 1, 1,
  1, 1, 0
), byrow = TRUE, ncol = 3)
colnames(A) <- letters[1:ncol(A)]
rownames(A) <- letters[(ncol(A) + 1):(nrow(A) + ncol(A))]
hypergraph(A, both = TRUE)
```

---

ind_rand_matrix *Independent random matrix*

Description

The function creates random matrices following a uniform probability over the space of networks having exactly m fixed number of edges (Moreno and Jennings, 1938; Rapoport, 1948; Solomonoff and Rapoport, 1951; Erdos and Renyi, 1959) or following a probability of the formation of the ties (Gilbert, 1959) assuming ties independency.

Usage

```
ind_rand_matrix(
  n,
  m = NULL,
  type = c("edges", "probability"),
  digraph = TRUE,
  loops = FALSE,
  l = NULL,
```

```
  p = NULL,
  trials = 1,
  multilevel = FALSE,
  sparse = FALSE
)
```

Arguments

n	The number of nodes of the first set
m	The number of nodes of a second set
type	The model assumes a fixed number of edges model (a.k.a. G(n,m)) (default) or a probability model (a.k.a. G(n,p))
digraph	Whether the matrix is symmetric or not
loops	Whether to expect nonzero elements in the diagonal of the matrix
l	The number of ties expected for the edges (a.k.a. G(n,m)) model
p	The probability of the ties expected for the probability (a.k.a. G(n,p)) model. If no parameter 'p' is specified, a uniform distribution is considered (p=0.5).
trials	Whether to add counting numbers to the probability (a.k.a. G(n,p)) model
multilevel	Whether to return a meta-matrix to represent a multilevel network
sparse	Whether to return a sparse matrix of the Matrix package, which is built without the dense matrix

Details

The fixed model is often called the G(n,m) graph with 'n' nodes and 'm' edges, and the 'm' edges are chosen uniformly randomly from the set of all possible ties.

The probability model is known as the G(n,p) graph, in which the matrix has 'n' nodes, and for each tie, the probability that it is present in the matrix is 'p'.

These are the simplest models that follow a conditional uniform distribution that place nonnull probability on a subset of networks with distinctive characteristics corresponding to the observed networks - for example, simulating a matrix based on the number of ties observed in the network.

With sparse = TRUE the ties are drawn among the cells that the model allows, and only they are stored, so the memory grows with the number of ties and not with the square of the number of nodes. The two models are the same as with a dense matrix: the number of ties of G(n,p) is binomial, which is what drawing every tie independently gives. It is available for one-mode and two-mode networks with trials = 1.

Value

This function returns a random matrix, or a list of matrices for a multilevel network.

Author(s)

Alejandro Espinosa-Rada

References

Erdos, P. and Renyi, A. (1959). On random graphs. Publicationes Mathematicae 6, 290–297.

Gilbert, N. (1959). Random Graphs. The Annals of Mathematical Statistics, 30(4): 1141-1144.

Moreno, J. and Jennings, H. (1938). Statistics of social configurations. Sociometry, 1(3/4):342–374.

Rapoport, A. (1948). Cycle distributions in random nets. Bulletin of Mathematical Biology, 10(3):145–157.

Solomonoff, R. and Rapoport, A. (1951). Connectivity of random nets. Bulletin of Mathematical Biology, 13:107–117.

Examples

```
set.seed(18051889)
ind_rand_matrix(5, type = "edges", l = 3, digraph = TRUE, loops = TRUE)
ind_rand_matrix(5, type = "probability")
ind_rand_matrix(n = 5, m = 2, p = 0.20, type = "probability", multilevel = TRUE)

# Large networks are cheaper as sparse matrices
dim(ind_rand_matrix(10000, type = "edges", l = 5000, digraph = FALSE, sparse = TRUE))
```

---

indirect_rel *Indirect relations*

Description

Relations between the nodes that are derived from the ties of the network, to be compared with `pos_dominance()`.

Usage

```
indirect_rel(
  A,
  type = c("distance", "adjacency", "walks", "shared"),
  digraph = TRUE,
  alpha = 0.1
)
```

Arguments

`A`	A square matrix
`type`	The relation: `adjacency`, `distance` (default), `walks` or `shared`
`digraph`	Whether the matrix is directed or undirected
`alpha`	The discount of the longer walks, for the `walks` relation

Details

`adjacency`: the ties themselves.

`distance`: the length of the shortest path between the nodes.

`walks`: the number of walks of any length that join two nodes, where a walk of length k is discounted by α^k, as in the Katz centrality.

`shared`: the number of neighbours that two nodes have in common.

Value

This function returns a square matrix of relations.

Author(s)

Alejandro Espinosa-Rada

References

Brandes, U. (2016). Network positions. Methodological Innovations, 9, 1–19. doi:10.1177/2059799116630650

Examples

```
A <- matrix(c(
  0, 1, 1, 1, 0,
  1, 0, 0, 0, 0,
  1, 0, 0, 0, 1,
  1, 0, 0, 0, 0,
  0, 0, 1, 0, 0
), byrow = TRUE, ncol = 5)
rownames(A) <- letters[1:nrow(A)]
colnames(A) <- rownames(A)

indirect_rel(A, type = "distance", digraph = FALSE)
indirect_rel(A, type = "walks", digraph = FALSE)
```

---

jaccard *Jaccard similarity*

Description

Jaccard similarity identifies the changes of ties between two matrices.

Usage

```
jaccard(
  A,
  B,
  directed = TRUE,
  diag = FALSE,
  coparticipation = FALSE,
  bipartite = FALSE
)
```

Arguments

A	Binary matrix A
B	Binary matrix B
directed	Whether the matrix is directed (asymmetric)
diag	Whether the diagonal should be considered
coparticipation	Select nodes that co-participate in both matrices
bipartite	Whether the matrix is incidence

Value

The output are: jaccard = Jaccard similarity, proportion = proportion among the ties present at a given observation of ties that are also present in the other matrix, and table = a table with the tie changes between matrices.

If coparticipation = TRUE, then also: match = The number of nodes present in both matrices; size_matrix1 = The size of the first matrix; size_matrix2 = The size of the second matrix; coparticipation1 = The percentage of nodes in the first matrix also present in the second matrix; coparticipation2 = The percentage of nodes in the second matrix also present in the first matrix: overlap_actors = Overlap of nodes between two matrices

If coparticipation = TRUE and bipartite = TRUE, then also: matchM1 = The number of nodes in the first 'mode' present in both matrices; matchM2 = The number of nodes in the second 'mode' present in both matrices; size_matrix1_M1 = The number of nodes in the first 'mode' of the first matrix; size_matrix1_M2 = The number of nodes in the second 'mode' of the first matrix; size_matrix2_M1 = The number of nodes in the first 'mode' of the second matrix; size_matrix2_M2 = The number of nodes in the second 'mode' of the second matrix; coparticipation1_M2 = The percentage of nodes of the first 'mode' in the first matrix present in the second matrix. coparticipation1_M2 = The percentage of nodes of the second 'mode' in the first matrix present in the second matrix. coparticipation2_M1 = The percentage of nodes of the first 'mode' in the second matrix present in the first matrix. coparticipation2_M2 = The percentage of nodes of the second 'mode' in the second matrix present in the first matrix. overlap_actors_M1 = Overlap between two matrices (nodes of the first 'mode') overlap_actors_M2 = Overlap between two matrices (nodes of the second 'mode')

Author(s)

Alejandro Espinosa-Rada

References

Batagelj, V., and Bren, M. (1995). Comparing resemblance measures. Journal of Classification 12, 73–90.

Examples

```
A <- matrix(c(
  0, 1, 1, 0,
  1, 0, 0, 0,
  1, 0, 0, 0,
  0, 0, 1, 0
), byrow = TRUE, ncol = 4)
B <- matrix(c(
  0, 1, 1, 0,
  1, 0, 0, 0,
  1, 0, 0, 0,
  0, 0, 0, 0
), byrow = TRUE, ncol = 4)
jaccard(A, B, directed = TRUE)
```

---

k_core *Generalized k-core*

Description

Generalized k-core for undirected, directed, weighted and multilevel networks

Usage

```
k_core(
  A,
  B1 = NULL,
  multilevel = FALSE,
  type = "in",
  digraph = FALSE,
  loops = FALSE,
  weighted = FALSE,
  alpha = 1
)
```

Arguments

A	A matrix object.
B1	An incidence matrix for multilevel networks.
multilevel	Whether the measure of k-core is for multilevel networks.
type	Character string, “out” (outdegree), “in” (indegree) and “all” (degree)
digraph	Whether the matrix is directed or undirected

loops	Whether the diagonal of the matrix is considered or not
weighted	Whether the measure of k-core is for valued matrices
alpha	Sets the alpha parameter in the generalised measures from Opsahl et al. (2010)

Details

For a binary network (weighted = FALSE and multilevel = FALSE), the coreness of a node is the largest k such that the node belongs to a subgraph in which every node has at least k ties (Seidman, 1983). It is obtained by removing, for k = 0, 1, 2, ..., the nodes with at most k ties among the remaining nodes (Batagelj and Zaversnik, 2011). A value larger than one counts as several ties, and an undirected network uses the tie in either direction. The loops are counted only when loops = TRUE, twice for an undirected network, as in igraph::coreness.

Value

This function return the k-core.

Author(s)

Alejandro Espinosa-Rada

References

Batagelj, V., & Zaveršnik, M. (2011). Fast algorithms for determining (generalized) core groups in social networks. Advances in Data Analysis and Classification, 5(2), 129–145. doi:10.1007/s116340100079y

Eidsaa, M., & Almaas, E. (2013). s-core network decomposition: A generalization of k-core analysis to weighted networks. Physical Review E, 88(6), 062819. doi:10.1103/PhysRevE.88.062819

Seidman S (1983). 'Network structure and minimum degree'. Social Networks, 5, 269-287.

Examples

```
A1 <- matrix(c(
  0, 1, 0, 0, 0,
  1, 0, 0, 1, 0,
  0, 0, 0, 1, 0,
  0, 1, 1, 0, 1,
  0, 0, 0, 1, 0
), byrow = TRUE, ncol = 5)
B1 <- matrix(c(
  1, 0, 0,
  1, 1, 0,
  0, 1, 0,
  0, 1, 0,
  0, 1, 1
), byrow = TRUE, ncol = 3)

k_core(A1, B1, multilevel = TRUE)
```

`katz_centrality` *Katz centrality*

Description

Katz centrality (1953), which counts the paths that arrive to a node, discounting the longer ones.

Usage

```
katz_centrality(
  A,
  alpha = 0.1,
  beta = 1,
  type = c("in", "out"),
  digraph = TRUE,
  weighted = FALSE
)
```

Arguments

`A`	A square matrix
`alpha`	The attenuation of the longer paths
`beta`	The status that every node has independently of the network
`type`	Whether the paths considered are the ones that arrive (`in`, default) or leave (`out`) the node
`digraph`	Whether the matrix is directed or undirected
`weighted`	Whether the matrix is weighted

Details

The centrality is $x = \beta(I - \alpha A^T)^{-1}1$, so a path of length k contributes α^k. The attenuation `alpha` should be smaller than the inverse of the leading eigenvalue of the matrix, otherwise the sum does not converge.

Value

This function returns the Katz centrality of the nodes.

Author(s)

Alejandro Espinosa-Rada

References

Katz, L. (1953). A new status index derived from sociometric analysis. Psychometrika, 18(1), 39–43. doi:10.1007/BF02289026

Examples

```
A <- matrix(c(
  0, 1, 1, 1, 0,
  1, 0, 0, 0, 0,
  1, 0, 0, 0, 1,
  1, 0, 0, 0, 0,
  0, 0, 1, 0, 0
), byrow = TRUE, ncol = 5)
rownames(A) <- letters[1:nrow(A)]
colnames(A) <- rownames(A)

katz_centrality(A, alpha = 0.1, digraph = FALSE)
```

---

kp_reciprocity *Reciprocity of Katz and Powell*

Description

Reciprocity of Katz and Powell

Usage

```
kp_reciprocity(G, fixed = FALSE, d = NULL, dichotomic = TRUE)
```

Arguments

G	A symmetric matrix object.
fixed	Whether the choices are fixed or not
d	Numeric value of the number of fixed choices.
dichotomic	Whether the matrix is weighted or binary

Value

This function gives a measurement of the tendency toward reciprocation of choices.

Author(s)

Alejandro Espinosa-Rada

References

Katz, L. and Powell, J.H. (1955). "Measurement of the tendency toward reciprocation of choice." Sociometry, 18:659-665.

Examples

```
data(krackhardt_friends)
kp_reciprocity(krackhardt_friends, fixed = TRUE, d = 5)
```

krackhardt_friends *Krackhardt friends*

Description

Friendship network of the relations measured for Krackhardt's high-tech managers.

Usage

```
data(krackhardt_friends)
```

Format

A 21 x 21 directed matrix of the managers

Source

Wasserman, S. and Faust, K. (1994). Social network analysis: Methods and applications. Cambridge University Press.

krackhardt_index *Krackhardt's dimensions of informal organisations*

Description

The four dimensions that Krackhardt (1994) uses to compare a network with a perfect hierarchy (an out-tree): connectedness, hierarchy, efficiency and least upper boundedness.

Usage

```
krackhardt_index(A, lubness = c("upper", "least"))
```

Arguments

A	A square matrix
lubness	Whether every pair of nodes should have an `upper` bound (default, Everett and Krackhardt, 2012) or a `least` upper bound (Krackhardt, 1994)

Details

The measures are computed on the reachability matrix R, where $R[i, j] = 1$ when j can be reached from i:

`connectedness` is the proportion of pairs of nodes that are connected in the underlying graph, i.e. one minus the proportion of pairs in different weak components.

`hierarchy` is one minus the proportion of the reachable ordered pairs that are also reachable in the opposite direction. It is one when no pair of nodes can reach each other.

`efficiency` is one minus the proportion of the ties that are not needed to keep the same weak components. A network is efficient when it has no more ties than a spanning tree.

`lubness` (upper boundedness) is the proportion of the pairs of nodes that have an upper bound, i.e. a node that reaches both of them. Everett and Krackhardt (2012) recommend this version, as the original condition asks for a *least* upper bound, an upper bound that is on a directed path from every other upper bound to both nodes, which need not be unique and can be a very distant node. The original condition is used with `lubness = "least"`. In both cases a node reaches itself, and the violations are counted within the weak components of more than two nodes.

All the measures are one for a perfect out-tree.

Value

This function returns the connectedness, hierarchy, efficiency and least upper boundedness of the network.

Author(s)

Alejandro Espinosa-Rada

References

Everett, M. G. and Krackhardt, D. (2012). A second look at Krackhardt's graph theoretical dimensions of informal organizations. Social Networks, 34(2), 159–163. doi:10.1016/j.socnet.2011.10.006

Krackhardt, D. (1994). Graph theoretical dimensions of informal organizations. In K. M. Carley and M. J. Prietula (Eds.), Computational Organization Theory (pp. 89–111). Hillsdale, NJ: Lawrence Erlbaum.

Examples

```
# A perfect out-tree
A <- matrix(c(
  0, 1, 1, 0, 0, 0, 0,
  0, 0, 0, 1, 1, 0, 0,
  0, 0, 0, 0, 0, 1, 1,
  0, 0, 0, 0, 0, 0, 0,
  0, 0, 0, 0, 0, 0, 0,
  0, 0, 0, 0, 0, 0, 0,
  0, 0, 0, 0, 0, 0, 0
), byrow = TRUE, ncol = 7)
rownames(A) <- letters[1:nrow(A)]
colnames(A) <- rownames(A)
```

```
krackhardt_index(A)
krackhardt_index(A, lubness = "least")
```

`lazega_lawfirm` *Lazega law firm*

Description

The data is part of a study carried out in a Northeastern US corporate law firm, referred to as SG&R, 1988-1991 in New England. The data were collected by Emmanuel Lazega (2001). This is a multiplex network of attorneys (partners and associates) of this firm. It includes (among others) measurements of networks among the 71 attorneys (partners and associates) of this firm, i.e. their strong-coworker network, advice network, friendship network, and indirect control networks.

Usage

```
data(lazega_lawfirm)
```

Format

Three 71 X 71 matrices:

cowork A matrix indicating "cowork" relationships among the attorneys, based on their work together on cases and other professional activities.

advice A matrix indicating "advice" relationships, where the matrix shows to whom attorneys went for professional advice.

friends A matrix indicating "friends" relationships, showing social connections outside of work.

attributes A data frame with attributes of the actors, including:

seniority Seniority.

status 1=partner; 2=associate

gender 1=man; 2=woman

office 1=Boston; 2=Hartford; 3=Providence

years Years with the firm

age Age

practice 1=litigation; 2=corporate

law_school 1 = Harvard, Yale; 2 = UConn; 3 = Other

Source

Lazega, Emmanuel (2001) The Collegial Phenomenon: The Social Mechanisms of Cooperation Among Peers in a Corporate Law Partnership. Oxford University Press.

`leading_eigen` *Communities with the leading eigenvector*

Description

Groups of nodes that are connected among themselves more often than expected, found with the leading eigenvector of the modularity matrix (Newman, 2006).

Usage

```
leading_eigen(A, digraph = FALSE, weighted = FALSE, max_groups = NULL)
```

Arguments

`A`	A square matrix
`digraph`	Whether the matrix is directed or undirected
`weighted`	Whether the matrix is weighted
`max_groups`	Maximum number of groups

Details

The modularity matrix is $B = A - kk^T/2m$, the observed ties minus the ties expected from the degrees. The sign of its leading eigenvector splits the network into two groups, and each group is split again while the modularity of the partition increases.

Value

This function returns the group of each node, the number of groups and the modularity of the partition.

Author(s)

Alejandro Espinosa-Rada

References

Newman, M. E. J. (2006). Finding community structure in networks using the eigenvectors of matrices. Physical Review E, 74(3), 036104. doi:10.1103/PhysRevE.74.036104

Examples

```
A <- matrix(c(
  0, 1, 1, 0, 0, 0,
  1, 0, 1, 0, 0, 0,
  1, 1, 0, 1, 0, 0,
  0, 0, 1, 0, 1, 1,
  0, 0, 0, 1, 0, 1,
  0, 0, 0, 1, 1, 0
```

```
), byrow = TRUE, ncol = 6)
rownames(A) <- letters[1:nrow(A)]
colnames(A) <- rownames(A)

leading_eigen(A)
```

`leiden` *Leiden communities*

Description

Communities found with the algorithm of Traag, Waltman and van Eck (2019), which improves the algorithm of Louvain by guaranteeing that the groups it returns are internally connected.

Usage

```
leiden(
  A,
  resolution = 1,
  objective = c("modularity", "cpm"),
  theta = 0.01,
  iterations = 10,
  weighted = FALSE,
  refine = TRUE
)
```

Arguments

`A`	A symmetric matrix object
`resolution`	Resolution of the quality function. Higher values give smaller groups
`objective`	Quality function: `modularity` (default) or `cpm`
`theta`	How random the refinement is. Small values only accept the best merges
`iterations`	Maximum number of times the three phases are repeated
`weighted`	Whether the matrix is weighted
`refine`	Whether the groups are refined before the aggregation. Without the refinement the algorithm is the one of Louvain (Blondel et al., 2008), which can return groups that are internally disconnected

Details

The algorithm has three phases that are repeated until the partition no longer changes. First, every node is moved to the neighbouring group that improves the quality of the partition the most, while any move improves it. Second, each group is refined: within the group, the nodes start as singletons and are merged at random among the merges that improve the quality, which is what keeps the groups connected. Third, the refined groups become the nodes of an aggregated network, and the process starts again on it.

Two quality functions are available. The `modularity` compares the ties within groups with the ties expected from the degrees, and suffers from a resolution limit. The constant Potts model (`cpm`) compares them with a constant density given by `resolution`, so the groups it finds do not depend on the size of the network.

The refinement is random, so the result depends on the seed.

Value

This function returns the group of each node, the number of groups, and the modularity and the quality of the partition.

Author(s)

Alejandro Espinosa-Rada

References

Blondel, V. D., Guillaume, J.-L., Lambiotte, R. and Lefebvre, E. (2008). Fast unfolding of communities in large networks. Journal of Statistical Mechanics, 2008(10), P10008. doi:10.1088/1742-5468/2008/10/P10008

Traag, V. A., Waltman, L. and van Eck, N. J. (2019). From Louvain to Leiden: guaranteeing well-connected communities. Scientific Reports, 9, 5233. doi:10.1038/s4159801941695z

Examples

```
A <- matrix(0, 12, 12)
A[1:4, 1:4] <- 1
A[5:8, 5:8] <- 1
A[9:12, 9:12] <- 1
diag(A) <- 0
A[4, 5] <- 1
A[5, 4] <- 1
A[8, 9] <- 1
A[9, 8] <- 1
rownames(A) <- letters[1:12]
colnames(A) <- rownames(A)

set.seed(18051889)
leiden(A)
```

---

main_path *Main path extraction from a citation network*

Description

Extracts the main path or key-route network from a directed citation adjacency matrix using traversal weights (SPC, SPLC, or SPNP).

Usage

```
main_path(
  A,
  weights = NULL,
  method = c("global", "local", "key_route"),
  weight_type = c("spc", "splc", "spnp"),
  k = 1L,
  seeds = NULL
)
```

Arguments

A	A square, named, directed adjacency matrix in which A[i,j] > 0 means that paper j cites paper i, so that the paths follow the flow of knowledge. See dag.
weights	Output of traversal_weights(). Computed internally with weight_type when NULL.
method	One of "global" (default), "local", or "key_route".
weight_type	One of "spc" (default), "splc", or "spnp". Ignored when weights is supplied.
k	Integer number of seed routes for method = "key_route". Default 1L.
seeds	Character vector of seed node names for method = "local".

Details

Three extraction strategies are available:

"global" Finds the single source-to-sink path that maximises the total accumulated edge weight, using dynamic programming along the topological order.

"local" Traces backward and forward from each node in seeds, always following the edge with the highest weight. Returns one route per seed.

"key_route" Ranks all edges by weight descending; for each of the top k seed edges not yet covered by a previous route, traces backward from the edge's tail and forward from the edge's head. The union of all routes forms a sub-DAG capturing multiple intellectual trajectories (Liu & Lu, 2012).

Value

A named list:

nodes Character vector of node names on the main path or key-route network (union across all routes).

edges Square adjacency matrix restricted to path nodes and edges, with the same values as A for included edges and zero elsewhere.

routes List of character vectors, one per extracted route, each giving the ordered sequence of node names.

weights The traversal_weights() result used.

Author(s)

Alejandro Espinosa-Rada

References

Hummon, N.P. and Doreian, P. (1989). Connectivity in a citation network: The development of DNA theory. Social Networks. 11(1): 39-63. doi:10.1016/03788733(89)900178.

Liu, J.S. and Lu, L.Y.Y. (2012). An integrated approach for main path analysis: Development of the Hirsch index as an example. Journal of the American Society for Information Science and Technology. 63(3): 528-542. doi:10.1002/asi.21692.

Lucio-Arias, D. and Leydesdorff, L. (2008). Main-path analysis and path-dependent transitions in HistCite-based historiographs. Journal of the American Society for Information Science and Technology. 59(12): 1948-1962. doi:10.1002/asi.20903.

Verspagen, B. (2007). Mapping technological trajectories as patent citation networks. Advances in Complex Systems. 10(1): 93-115. doi:10.1142/S0219525907000945.

Examples

```
A <- matrix(c(
  0, 1, 1, 0, 0, 0,
  0, 0, 0, 1, 1, 0,
  0, 0, 0, 0, 1, 0,
  0, 0, 0, 0, 0, 0,
  0, 0, 0, 0, 0, 1,
  0, 0, 0, 0, 0, 0
), byrow = TRUE, nrow = 6)
rownames(A) <- letters[1:6]
colnames(A) <- letters[1:6]

mp <- main_path(A, method = "global")
mp$routes

kr <- main_path(A, method = "key_route", k = 2L)
kr$nodes
```

---

main_path_diag *Diagnostics for main path analysis*

Description

Computes K-sensitivity of the key-route network and the Jaccard overlap between SPC- and SPLC-based key routes as a robustness check.

Usage

```
main_path_diag(
  A,
  weights = NULL,
  k_values = c(5L, 10L, 15L, 20L, 30L),
  k_jaccard = 10L
)
```

Arguments

A	A square, named, directed adjacency matrix in which A[i,j] > 0 means that paper j cites paper i.
weights	Output of traversal_weights() with method = "spc". Computed internally when NULL.
k_values	Integer vector of K values for the sensitivity table. Default c(5, 10, 15, 20, 30).
k_jaccard	Integer K used for the SPLC/SPC Jaccard comparison. Default 10L.

Details

The K-sensitivity table shows how the size of the key-route network grows as more seed routes are added. Stabilisation of new-node counts signals that the main structural backbone has been captured.

The SPLC/SPC Jaccard overlap at k_jaccard routes measures whether the key routes change when the intermediate papers are also counted as origins of knowledge (SPLC) instead of only the sources (SPC), which is the main difference between the two weights (Liu, Lu and Ho, 2019). A value of one means that both weights give the same papers.

Value

A named list:

k_sensitivity Data frame with columns K, n_nodes, n_edges, and new_nodes (marginal nodes added at each K).

splc_spc_jaccard Numeric Jaccard overlap of node sets between SPLC- and SPC-based key routes at k_jaccard.

weight_summary Summary statistics for nonzero edge weights.

n_sources Number of source nodes.

n_sinks Number of sink nodes.

total_log_paths Log of the total number of search paths of weights.

Author(s)

Alejandro Espinosa-Rada

References

Liu, J.S., Lu, L.Y.Y. and Ho, M.H.C. (2019). A few notes on main path analysis. Scientometrics. 119(1): 379-391. doi:10.1007/s1119201903034x.

Verspagen, B. (2007). Mapping technological trajectories as patent citation networks. Advances in Complex Systems. 10(1): 93-115. doi:10.1142/S0219525907000945.

Examples

```
A <- matrix(c(
  0, 1, 1, 0, 0, 0,
  0, 0, 0, 1, 1, 0,
  0, 0, 0, 0, 1, 0,
  0, 0, 0, 0, 0, 0,
  0, 0, 0, 0, 0, 1,
  0, 0, 0, 0, 0, 0
), byrow = TRUE, nrow = 6)
rownames(A) <- letters[1:6]
colnames(A) <- letters[1:6]

main_path_diag(A, k_values = c(1L, 2L, 3L))
```

---

matrix_adjlist *Transform a matrix to an adjacency list*

Description

Transform a matrix to an adjacency list

Usage

```
matrix_adjlist(A)
```

Arguments

A	A matrix

Value

This function transform a matrix to an adjacency list

Author(s)

Alejandro Espinosa-Rada

Examples

```
A <- matrix(c(
  0, 1, 1, 0, 0, 0, 0, 1, 0,
  1, 0, 1, 0, 0, 0, 0, 0, 0,
  1, 1, 0, 0, 0, 0, 0, 0, 0,
  0, 0, 0, 0, 1, 1, 0, 0, 0,
  0, 0, 0, 1, 0, 0, 0, 0, 0,
  0, 0, 0, 1, 0, 0, 1, 1, 0,
  0, 0, 0, 0, 0, 1, 0, 1, 0,
  1, 0, 0, 0, 0, 1, 1, 0, 0,
  0, 0, 0, 0, 0, 0, 0, 0, 0
), byrow = TRUE, ncol = 9)
rownames(A) <- letters[1:nrow(A)]
colnames(A) <- rownames(A)
matrix_adjlist(A)
```

`matrix_projection` *Unipartite projections*

Description

Two-mode networks can be represented (or 'projected') as one-mode networks.

Usage

```
matrix_projection(A, B = NULL, digraph = FALSE)
```

Arguments

`A`	A first matrix object
`B`	A second matrix object
`digraph`	Whether the matrix is directed or not

Value

This function return a list of matrices of the two projections of the original matrix.

Author(s)

Alejandro Espinosa-Rada

References

Davis, Allison; Gardner, Burleigh B. and Mary. R. Gardner (1941). Deep South: A Social Anthropological Study of Caste and Class. The University of Chicago Press, Chicago.

Breiger, Ronald L. (1976). The Duality of Persons and Groups, 53(2), 181-190 doi:10.2307/2576011

Wasserman, S. and Faust, K. (1994). Social network analysis: Methods and applications. Cambridge University Press.

Examples

```
A <- matrix(c(
  2, 0, 2,
  1, 1, 0,
  0, 3, 3,
  0, 2, 2,
  0, 0, 1
), byrow = TRUE, ncol = 3)
matrix_projection(A)

A <- matrix(c(
  0, 0, 0, 0, 1,
  1, 0, 0, 0, 0,
  1, 1, 0, 0, 0,
  0, 1, 1, 1, 1,
  0, 0, 1, 0, 0,
  0, 0, 1, 1, 0
), byrow = TRUE, ncol = 5)

B <- matrix(c(
  0, 0, 0, 0, 1,
  1, 0, 0, 0, 0,
  1, 0, 0, 0, 0,
  0, 1, 0, 0, 0,
  0, 0, 1, 0, 0,
  0, 0, 1, 0, 0
), byrow = TRUE, ncol = 5)
matrix_projection(A, B, digraph = TRUE)
```

---

matrix_report *Matrix report*

Description

The primary matrix used in social network analysis are the adjacency matrix or sociomatrix, and the incidence matrix.

Usage

```
matrix_report(A)
```

Arguments

A	A matrix

Value

This function return a report of some of the characteristics of the matrix.

Author(s)

Alejandro Espinosa-Rada

References

Wasserman, S. and Faust, K. (1994). Social network analysis: Methods and applications. Cambridge University Press.

Examples

```
A <- matrix(c(
  1, 1, 0, 0, -1,
  1, 0, 0, 1, 1,
  0, 0, NA, 1, 1,
  0, 1, 1, 0, 1,
  1, 1, 1, 1, 0
), byrow = TRUE, ncol = 5)

B <- matrix(c(
  1, 0, 0,
  1, 1, 0,
  0, NA, 0,
  0, 1, 0,
  0, 1, 1
), byrow = TRUE, ncol = 3)
matrix_report(A)
matrix_report(B)
```

---

matrix_to_edgelist *Transform a square matrix to an edge-list*

Description

Transform a square matrix to an edge-list

Usage

```
matrix_to_edgelist(A, digraph = FALSE, valued = FALSE, loops = FALSE)
```

Arguments

A	A square matrix
digraph	Whether the matrix is directed or not
valued	Add a third columns with the valued of the relationship
loops	Whether the loops are retained or not

Value

This function transform the matrix into an edgelist

Author(s)

Alejandro Espinosa-Rada

Examples

```
A <- matrix(c(
  0, 2, 1,
  1, 0, 0,
  1, 0, 1
), byrow = TRUE, ncol = 3)
matrix_to_edgelist(A, digraph = TRUE, valued = TRUE, loops = TRUE)
```

---

meta_matrix *Meta matrix for multilevel networks*

Description

Meta matrix for multilevel networks

Usage

```
meta_matrix(A1, B1, A2 = NULL, B2 = NULL, A3 = NULL, B3 = NULL)
```

Arguments

A1	The square matrix of the lowest level
B1	The incidence matrix of the ties between the nodes of first level and the nodes of the second level
A2	The square matrix of the second level
B2	The incidence matrix of the ties between the nodes of the second level and the nodes of the third level
A3	The square matrix of the third level
B3	The incidence matrix of the ties between the nodes of the third level and the nodes of the first level

Details

The meta matrix places the network of each level on the diagonal and the ties between two levels in both triangles, as an incidence matrix and its transpose:

$$\begin{pmatrix} A_1 & B_1 & B_3^T \\ B_1^T & A_2 & B_2 \\ B_3 & B_2^T & A_3 \end{pmatrix}$$

The matrices that are not given are zero. The names of the nodes are kept when every level has them.

Value

Return a meta matrix for multilevel networks

Author(s)

Alejandro Espinosa-Rada

References

Carley, K. M. (2002). Smart agents and organizations of the future. In: Leah Lievrouw & Sonia Livingstone (Eds.), The Handbook of New Media (pp. 206-220). Thousand Oaks, CA, Sage.

Krackhardt, D., & Carley, K. M. (1998). PCANS model of structure in organizations (pp. 113-119). Pittsburgh, Pa, USA: Carnegie Mellon University, Institute for Complex Engineered Systems.

Examples

```
A1 <- matrix(c(
  0, 1, 0, 0, 0,
  1, 0, 0, 1, 0,
  0, 0, 0, 1, 0,
  0, 1, 1, 0, 1,
  0, 0, 0, 1, 0
), byrow = TRUE, ncol = 5)

B1 <- matrix(c(
  1, 0, 0,
  1, 1, 0,
  0, 1, 0,
  0, 1, 0,
  0, 1, 1
), byrow = TRUE, ncol = 3)

A2 <- matrix(c(
  0, 1, 1,
  1, 0, 0,
  1, 0, 0
), byrow = TRUE, nrow = 3)

B2 <- matrix(c(
  1, 1, 0, 0,
  0, 0, 1, 0,
  0, 0, 1, 1
), byrow = TRUE, ncol = 4)

A3 <- matrix(c(
  0, 1, 1, 1,
  1, 0, 0, 0,
  1, 0, 0, 1,
  1, 0, 1, 0
), byrow = TRUE, ncol = 4)
```

```
B3 <- matrix(c(
  1, 0, 0, 0, 0,
  0, 1, 0, 1, 0,
  0, 0, 0, 0, 0,
  0, 0, 0, 0, 0
), byrow = TRUE, ncol = 5)

rownames(A1) <- letters[1:nrow(A1)]
colnames(A1) <- rownames(A1)
rownames(A2) <- letters[nrow(A1) + 1:nrow(A2)]
colnames(A2) <- rownames(A2)
rownames(B1) <- rownames(A1)
colnames(B1) <- colnames(A2)
rownames(A3) <- letters[nrow(A1) + nrow(A2) + 1:nrow(A3)]
colnames(A3) <- rownames(A3)
rownames(B2) <- rownames(A2)
colnames(B2) <- colnames(A3)
rownames(B3) <- rownames(A3)
colnames(B3) <- rownames(A1)
meta_matrix(A1, B1, A2, B2, A3, B3)
```

`minmax_overlap` *Minimum/maximum overlap*

Description

Two-mode networks can be represented (or 'projected') as one-mode networks.

Usage

```
minmax_overlap(A, row = TRUE, min = TRUE)
```

Arguments

`A`	A matrix object
`row`	Whether to consider the actors in the rows of the matrix (default) or the column.
`min`	Whether to extract the minimum (default) or the maximum overlap.

Value

This function return the overlap between the modes (a.k.a. actors, nodes, vertices).

Author(s)

Alejandro Espinosa-Rada

References

Morris, S.A. (2005). Unified Mathematical Treatment of Complex Cascaded Bipartite Networks: The Case of Collections of Journal Papers. Unpublished PhD Thesis, Oklahoma State University.

Examples

```
A <- matrix(c(
  2, 0, 2,
  1, 1, 0,
  0, 3, 3,
  0, 2, 2,
  0, 0, 1
), byrow = TRUE, ncol = 3)
minmax_overlap(A)
```

`mix_matrix` *Mixing matrix*

Description

Create a mixing matrix from node attributes. The mixing matrix is a two-dimensional matrix that cross-classifies the edges depending on the values of their attributes. This matrix allowed identifying segregation and homophily at the network level.

Usage

```
mix_matrix(A, att = NULL, digraph = TRUE)
```

Arguments

`A`	A symmetric matrix object
`att`	Categorical attribute of the nodes
`digraph`	Whether the matrix is directed. A symmetric matrix is always treated as undirected

Details

Values in the diagonal are the number of ties within groups, and off-diagonal are the number of relations between groups. For directed networks the entries are the arcs between the groups. For undirected networks every edge is counted once, so the entries outside the diagonal are half of the edges between the two groups.

Value

This function returns a mixing matrix

Author(s)

Alejandro Espinosa-Rada

Examples

```
n <- 100
A <- matrix(c(rbinom(n, 1, 0.5)),
  ncol = sqrt(n), nrow = sqrt(n), byrow = TRUE
)
rownames(A) <- letters[1:nrow(A)]
colnames(A) <- letters[1:ncol(A)]
att <- rbinom(sqrt(n), 3, 0.5)
mix_matrix(A, att = att)
```

`mixed_census` *Multilevel triad and quadrilateral census*

Description

Multilevel triad and quadrilateral census

Usage

```
mixed_census(A1, B1, B2 = NULL, quad = FALSE)
```

Arguments

A1	An adjacent matrix object.
B1	An incidence matrix object.
B2	An incidence matrix object.
quad	Whether the matrix is a quadrilateral census or not.

Value

This function return the counts of a multilevel census.

If `quad = TRUE`, then the function return the multilevel quadrilateral census.

Author(s)

Alejandro Espinosa-Rada

References

Espinosa-Rada, A. (2021). A Network Approach for the Sociological Study of Science: Modelling Dynamic Multilevel Networks. [PhD](https://research.manchester.ac.uk/en/studentTheses/a-network-approach-for-the-sociological-study-of-science-and-know). The University of Manchester.

Espinosa-Rada, A., Bellotti, E., Everett, M., & Stadtfeld, C. (2024). Co-evolution of a socio-cognitive scientific network: A case study of citation dynamics among astronomers. Social Networks, 78, 92–108. doi:10.1016/j.socnet.2023.11.008

Hollway, J., Lomi, A., Pallotti, F., & Stadtfeld, C. (2017). Multilevel social spaces: The network dynamics of organizational fields. Network Science, 5(2), 187–212. doi:10.1017/nws.2017.8

Examples

```
B1 <- matrix(c(
  1, 1, 0,
  0, 0, 1,
  0, 0, 1,
  1, 0, 0
), byrow = TRUE, ncol = 3)
A1 <- matrix(c(
  0, 1, 0, 1,
  1, 0, 0, 1,
  0, 1, 0, 1,
  1, 0, 1, 0
), byrow = TRUE, ncol = 4)
B2 <- matrix(c(
  1, 0, 0, 0, 0,
  0, 1, 0, 1, 0,
  0, 0, 0, 0, 0,
  0, 0, 0, 0, 0
), byrow = TRUE, ncol = 5)

mixed_census(A1, B1, B2, quad = TRUE)
```

---

modularity_score *Modularity*

Description

Modularity of a partition of the nodes (Newman and Girvan, 2004): the proportion of the ties that are within groups, minus the proportion that would be expected if the ties were distributed at random keeping the degree of every node.

Usage

```
modularity_score(
  A,
  partition,
  method = c("newman", "linkrank"),
  digraph = FALSE,
  weighted = FALSE,
  resolution = 1,
  damping = 0.85
)
```

Arguments

A	A square matrix
partition	A vector with the group of each node
method	Whether to use the modularity of newman (default) or of linkrank

digraph	Whether the matrix is directed or undirected
weighted	Whether the matrix is weighted
resolution	Weight given to the expected ties. Values above one give smaller groups
damping	Probability of following a tie in the random walk of the linkrank method

Details

The modularity is positive when the nodes of a group are connected among themselves more often than expected. It is used to compare partitions of the same network, as its maximum depends on the network. For directed networks, the expected ties use the out-degree of the sender and the in-degree of the receiver (Arenas et al., 2007).

The linkrank method (Kim, Son and Jeong, 2010) replaces the ties by the flow of a random walker: the value of a tie is the PageRank of the sender times the probability that the walker uses that tie, and the expected value is the product of the PageRank of both nodes. It takes the direction of the ties into account, which the modularity of Arenas et al. does only through the degrees.

Modularity has a resolution limit (Fortunato and Barthelemy, 2007): it does not detect groups below a size that depends on the size of the network. The resolution parameter changes the weight of the expected ties to look for smaller or larger groups.

Value

This function returns the modularity of the partition.

Author(s)

Alejandro Espinosa-Rada

References

Arenas, A., Duch, J., Fernandez, A. and Gomez, S. (2007). Size reduction of complex networks preserving modularity. New Journal of Physics, 9(6), 176. doi:10.1088/13672630/9/6/176

Fortunato, S. and Barthelemy, M. (2007). Resolution limit in community detection. Proceedings of the National Academy of Sciences, 104(1), 36–41. doi:10.1073/pnas.0605965104

Kim, Y., Son, S.-W. and Jeong, H. (2010). Finding communities in directed networks. Physical Review E, 81(1), 016103. doi:10.1103/PhysRevE.81.016103

Newman, M. E. J. and Girvan, M. (2004). Finding and evaluating community structure in networks. Physical Review E, 69(2), 026113. doi:10.1103/PhysRevE.69.026113

Examples

```
A <- matrix(c(
  0, 1, 1, 0, 0, 0,
  1, 0, 1, 0, 0, 0,
  1, 1, 0, 1, 0, 0,
  0, 0, 1, 0, 1, 1,
  0, 0, 0, 1, 0, 1,
  0, 0, 0, 1, 1, 0
), byrow = TRUE, ncol = 6)
```

```
rownames(A) <- letters[1:nrow(A)]
colnames(A) <- rownames(A)

modularity_score(A, partition = c(1, 1, 1, 2, 2, 2))
modularity_score(A, partition = c(1, 1, 1, 2, 2, 2), method = "linkrank", digraph = TRUE)
```

`multilevel_degree` *Degree centrality for multilevel networks*

Description

Degree of the nodes of a network of two or three levels, counting the ties within each level and the ties between the levels in different combinations.

Usage

```
multilevel_degree(
  A1,
  B1,
  A2 = NULL,
  B2 = NULL,
  A3 = NULL,
  B3 = NULL,
  complete = FALSE,
  digraphA1 = FALSE,
  digraphA2 = FALSE,
  digraphA3 = FALSE,
  typeA1 = "out",
  typeA2 = "out",
  typeA3 = "out",
  loopsA1 = FALSE,
  loopsA2 = FALSE,
  loopsA3 = FALSE,
  normalized = FALSE,
  weightedA1 = FALSE,
  weightedA2 = FALSE,
  weightedA3 = FALSE,
  alphaA1 = 0.5,
  alphaA2 = 0.5,
  alphaA3 = 0.5
)
```

Arguments

A1	The square matrix of the lowest level
B1	The incidence matrix of the ties between the nodes of first level and the nodes of the second level

A2	The square matrix of the second level
B2	The incidence matrix of the ties between the nodes of the second level and the nodes of the third level
A3	The square matrix of the third level
B3	The incidence matrix of the ties between the nodes of the third level and the nodes of the first level
complete	Whether to return every column described in the details, instead of only the multilevel degree
digraphA1	Whether A1 is a directed network
digraphA2	Whether A2 is a directed network
digraphA3	Whether A3 is a directed network
typeA1	Type of degree of the network for A1, "out" for out-degree, "in" for in-degree or "all" for the sum of the two
typeA2	Type of degree of the network for A2, "out" for out-degree, "in" for in-degree or "all" for the sum of the two
typeA3	Type of degree of the network for A3, "out" for out-degree, "in" for in-degree or "all" for the sum of the two
loopsA1	Whether the loops of the edges are considered in matrix A1
loopsA2	Whether the loops of the edges are considered in matrix A2
loopsA3	Whether the loops of the edges are considered in matrix A3
normalized	Whether to divide each degree by the largest value it could take, as described in the details (Espinosa-Rada et al., 2021)
weightedA1	Whether A1 is weighted
weightedA2	Whether A2 is weighted
weightedA3	Whether A3 is weighted
alphaA1	The alpha parameter of A1 according to Opsahl et al (2010) for weighted networks. The value 0.5 is given by default.
alphaA2	The alpha parameter of A2 according to Opsahl et al (2010) for weighted networks. The value 0.5 is given by default.
alphaA3	The alpha parameter of A3 according to Opsahl et al (2010) for weighted networks. The value 0.5 is given by default.

Details

The levels are placed in a single meta-matrix. Level one has n nodes and the ties A1, level two has m nodes and the ties A2, and level three has k nodes and the ties A3. The incidence matrices join the levels: B1 the first with the second (n by m), B2 the second with the third (m by k), and B3 the third with the first (k by n). A level that is not given has no ties within it.

Each column of the result emphasises a different section of the meta-matrix:

multilevel: every node counts the ties within its own level and its ties with the other levels, i.e. A1 + B1 + B3 for the first level, B1 + A2 + B2 for the second, and B2 + A3 + B3 for the third.

`bipartiteB1`, `bipartiteB2` and `bipartiteB3`: the degree in each incidence matrix, i.e. only the ties between two levels.

`tripartiteB1B2`, `tripartiteB1B3`, `tripartiteB2B3` and `tripartiteB1B2B3`: the degree in the union of incidence matrices, i.e. only the ties between levels.

`low_multilevel` (`A1 + B1 + B2 + B3`), `meso_multilevel` (`B1 + A2 + B2 + B3`) and `high_multilevel` (`B1 + B2 + A3 + B3`): the ties within a single level, together with all the ties between levels. For the nodes of the emphasised level they are the same as `multilevel`: the first level in `low_multilevel`, the second in `meso_multilevel` and the third in `high_multilevel`.

The rows are named `n1, n2, ...` for the first level, `m1, m2, ...` for the second and `k1, k2, ...` for the third. Without `complete = TRUE`, only the `multilevel` column is returned.

With `normalized = TRUE`, each degree is divided by the largest value it could take: the other nodes of the same level plus the nodes of the levels it is tied to. For the `multilevel` column this is `(n - 1) + m` for the first level (`(n - 1) + m + k` when `B3` is given), `(m - 1) + n + k` for the second level (`(m - 1) + n` with two levels), and `(k - 1) + m` for the third level (`(k - 1) + m + n` when `B3` is given). The bipartite degrees are divided by the number of nodes of the other level (Borgatti and Everett, 1997). The normalized values are only defined for binary matrices. All the values are rounded to three decimals.

The ties within each level can be directed (`digraphA1`, `typeA1`, ...) and weighted (`weightedA1`, `alphaA1`, ...), in which case the degree of Opsahl et al. (2010) is used. The ties between levels are undirected.

Value

A data frame with one row for each node of every level, and the `multilevel` degree, or every column described in the details when `complete = TRUE`

Author(s)

Alejandro Espinosa-Rada

References

Borgatti, S. P., and Everett, M. G. (1997). Network analysis of 2-mode data. Social Networks, 19(3), 243–269.

Freeman, L. C. (1978). Centrality in social networks conceptual clarification. Social Networks, 1(3), 215–239.

Opsahl, T., Agneessens, F., and Skvoretz, J. (2010). Node centrality in weighted networks: Generalizing degree and shortest paths. Social Networks, 32(3), 245–251.

Examples

```
A1 <- matrix(c(
  0, 1, 0, 0, 0,
  1, 0, 0, 1, 0,
  0, 0, 0, 1, 0,
  0, 1, 1, 0, 1,
  0, 0, 0, 1, 0
), byrow = TRUE, ncol = 5)
```

```
B1 <- matrix(c(
  1, 0, 0,
  1, 1, 0,
  0, 1, 0,
  0, 1, 0,
  0, 1, 1
), byrow = TRUE, ncol = 3)

A2 <- matrix(c(
  0, 1, 1,
  1, 0, 0,
  1, 0, 0
), byrow = TRUE, nrow = 3)

B2 <- matrix(c(
  1, 1, 0, 0,
  0, 0, 1, 0,
  0, 0, 1, 1
), byrow = TRUE, ncol = 4)

A3 <- matrix(c(
  0, 1, 1, 1,
  1, 0, 0, 0,
  1, 0, 0, 1,
  1, 0, 1, 0
), byrow = TRUE, ncol = 4)

B3 <- matrix(c(
  1, 0, 0, 0, 0,
  0, 1, 0, 1, 0,
  0, 0, 0, 0, 0,
  0, 0, 0, 0, 0
), byrow = TRUE, ncol = 5)

multilevel_degree(A1, B1, A2, B2, A3, B3)

multilevel_degree(A1, B1, A2, B2, A3, B3, normalized = TRUE, complete = TRUE)
```

`multiplex_census` *Multiplex triad census*

Description

This function counts the different subgraphs of three nodes in a multiplex directed and undirected network.

Usage

```
multiplex_census(A, B, merge = c("none", "overlap"))
```

Arguments

A	A directed matrix object.
B	An undirected matrix object. A directed matrix is replaced by its underlying graph.
merge	Whether to merge the classes that give the same overlapped triad (overlap) or not (none, default).

Details

Each triple of nodes is classified by its type in the first (directed) network, one of the 16 types of the triad census (Holland and Leinhardt, 1976), and by the position of the edges of the second (undirected) network in that triad (Espinosa-Rada, 2021: Figure 12). Each type of the first network is drawn in fixed positions, bottom left, top and bottom right, and the edges of the second network are named by where they fall: 102a (bottom left to top), 102b (bottom left to bottom right) and 102c (top to bottom right); the two-paths by their centre, 201a (bottom left), 201b (bottom right) and 201c (top). Positions that are equivalent by the symmetry of the triad of the first network form a single class, such as 021U_102ac, as the two edges between the top and the bottom nodes are equivalent when both bottom nodes send a tie to the top one.

With merge = "overlap", the classes of the same type of the first network that give the same triad when both networks are overlapped are also merged, as most groups of Figure 12 do (for instance, 102_003-102a: an edge of the second network on a mutual tie of the first adds nothing to the overlapped triad).

The counts of each type of the first network add up to its triad census.

Up to version 1.0-3 the function added counts of the two networks instead of counting the triples of each class, so its results were wrong.

Value

This function gives the number of triples in each class, named by the type of the first network and the position of the edges of the second.

Author(s)

Alejandro Espinosa-Rada

References

Batagelj, V. and Mrvar, A. (2001). A subquadratic triad census algorithm for large sparse networks with small maximum degree. Social Networks, 23(3), 237–243. doi:10.1016/S03788733(01)00035-1

Espinosa-Rada, A. (2021). A Network Approach for the Sociological Study of Science: Modelling Dynamic Multilevel Networks. [PhD](https://research.manchester.ac.uk/en/studentTheses/a-network-approach-for-the-sociological-study-of-science-and-know). The University of Manchester.

Espinosa-Rada, A., Bellotti, E., Everett, M., & Stadtfeld, C. (2024). Co-evolution of a socio-cognitive scientific network: A case study of citation dynamics among astronomers. Social Networks, 78, 92–108. doi:10.1016/j.socnet.2023.11.008

Holland, P. W. and Leinhardt, S. (1976). Local structure in social networks. Sociological Methodology, 7, 1–45. doi:10.2307/270703

Examples

```
# SOAR
A <- matrix(
  c(
    0, 1, 1, 1, 1, 0, 0, 1, 1, 0, 1, 1,
    0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
    0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0,
    0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
    0, 1, 1, 1, 0, 0, 0, 1, 1, 0, 1, 1,
    0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0,
    0, 0, 0, 1, 0, 1, 0, 0, 1, 0, 0, 0,
    0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
    0, 1, 1, 1, 1, 0, 0, 1, 0, 0, 1, 1,
    0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0,
    0, 1, 0, 0, 1, 0, 0, 0, 1, 0, 0, 1,
    0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
  ),
  byrow = TRUE, ncol = 12
)

B <- matrix(
  c(
    0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0,
    0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1,
    0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
    0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0,
    1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0,
    0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0,
    0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0,
    0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
    1, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0,
    0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
    0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0,
    0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
  ),
  byrow = TRUE, ncol = 12
)

multiplex_census(A, B)
```

`neigh_inclusion` *Neighbourhood inclusion*

Description

Neighbourhood-inclusion preorder of an undirected network (Brandes, 2016; Schoch and Brandes, 2016).

Usage

```
neigh_inclusion(A, closed = TRUE, direction = c("dominated", "dominates"))
```

Arguments

A	A symmetric matrix object
closed	Whether the neighbourhood of the dominating node is closed (i.e. includes the node itself)
direction	Whether P[u, v] = 1 means that u is dominated by v (default, as in Schoch and Brandes, 2016) or that u dominates v

Details

A node u is dominated by a node v if the open neighbourhood of u is included in the closed neighbourhood of v, $N(u) \subseteq N[v]$. With closed = FALSE both neighbourhoods are open, $N(u) \subseteq N(v)$.

In matrix form, u is dominated by v when the number of shared neighbours equals the degree of u.

Value

This function returns a binary matrix P of the neighbourhood inclusion, where P[u, v] = 1 when the neighbours of u are also neighbours of v, in the direction asked for.

Author(s)

Alejandro Espinosa-Rada

References

Brandes, U. (2016). Network positions. Methodological Innovations, 9, 1–19. doi:10.1177/2059799116630650

Schoch, D. and Brandes, U. (2016). Re-conceptualizing centrality in social networks. European Journal of Applied Mathematics, 27(6), 971–985. doi:10.1017/S0956792516000401

Examples

```
A <- matrix(c(
  0, 1, 1, 1, 0, 0, 0, 0, 0,
  1, 0, 1, 1, 1, 0, 0, 0, 0,
  1, 1, 0, 1, 0, 1, 0, 0, 0,
  1, 1, 1, 0, 1, 1, 0, 0, 0,
  0, 1, 0, 1, 0, 1, 1, 0, 0,
  0, 0, 1, 1, 1, 0, 1, 0, 0,
  0, 0, 0, 0, 1, 1, 0, 1, 0,
```

```
  0, 0, 0, 0, 0, 0, 1, 0, 1,
  0, 0, 0, 0, 0, 0, 0, 1, 0
), byrow = TRUE, ncol = 9)
rownames(A) <- letters[1:nrow(A)]
colnames(A) <- rownames(A)

neigh_inclusion(A)
```

`page_rank_centrality` *PageRank centrality*

Description

PageRank of Brin and Page (1998): the probability that a random walker, who follows the ties and sometimes jumps to a random node, is found in each node.

Usage

```
page_rank_centrality(
  A,
  damping = 0.85,
  digraph = TRUE,
  weighted = FALSE,
  tol = 1e-10,
  max_iter = 1000
)
```

Arguments

`A`	A square matrix
`damping`	Probability of following a tie instead of jumping to a random node
`digraph`	Whether the matrix is directed or undirected
`weighted`	Whether the matrix is weighted
`tol`	Tolerance of the power iteration
`max_iter`	Maximum number of iterations

Details

At each step, the walker follows one of the outgoing ties of the node with probability `damping`, and jumps to a node chosen at random with probability `1 - damping`. The nodes without outgoing ties are treated as if they were connected to every node.

Value

This function returns the PageRank of the nodes, which adds up to one.

Author(s)

Alejandro Espinosa-Rada

References

Brin, S. and Page, L. (1998). The anatomy of a large-scale hypertextual Web search engine. Computer Networks and ISDN Systems, 30(1-7), 107–117. doi:10.1016/S01697552(98)00110X

Examples

```
A <- matrix(c(
  0, 1, 1, 1, 0,
  1, 0, 0, 0, 0,
  1, 0, 0, 0, 1,
  1, 0, 0, 0, 0,
  0, 0, 1, 0, 0
), byrow = TRUE, ncol = 5)
rownames(A) <- letters[1:nrow(A)]
colnames(A) <- rownames(A)

page_rank_centrality(A, digraph = FALSE)
```

---

pareto_dominance *Pareto dominance*

Description

Pareto-style dominance across several neighbourhood-inclusion relations (Espinosa-Rada, 2026).

Usage

```
pareto_dominance(
  inclusions,
  tau = length(inclusions),
  strict = NULL,
  direction = c("dominated", "dominates")
)
```

Arguments

`inclusions`	A list of binary matrices where `[u, v]` = 1 if u is included in v
`tau`	Minimum number of relations in which u should be weakly dominated by v
`strict`	An optional list of binary matrices, in the same order as `inclusions`, where `[u, v]` = 1 if the inclusion of u in v is strict. If NULL, the asymmetric criterion is used
`direction`	Whether `D[u, v]` = 1 means that u is `dominated` by v (default) or that u `dominates` v

Details

Given K inclusion matrices (e.g. from `set_inclusion()`), u is dominated by v if (i) the neighbourhood of u is included in the neighbourhood of v in at least `tau` relations (weak dominance), and (ii) the dominance is strict in at least one relation.

By default, the dominance is strict when it is asymmetric: the neighbourhood of u is included in the neighbourhood of v but not the other way around. Other criteria, such as proper inclusion, can be given in `strict` (see `set_inclusion(proper = TRUE)`).

With `tau = K` every relation should agree (unanimity), with a majority of relations the dominance is less demanding, and with `tau = 1` one relation is enough.

Value

This function returns a binary dominance matrix `D`.

Author(s)

Alejandro Espinosa-Rada

References

Espinosa-Rada, A. (2026). Network positions within scholars and intellectual networks. Journal of Informetrics, 20, 101854. doi:10.1016/j.joi.2026.101854

Schoch, D. and Brandes, U. (2016). Re-conceptualizing centrality in social networks. European Journal of Applied Mathematics, 27(6), 971–985. doi:10.1017/S0956792516000401

Examples

```
X <- matrix(c(
  1, 1, 1, 0,
  1, 1, 0, 0,
  1, 0, 0, 0,
  0, 1, 0, 0,
  0, 0, 1, 1
), byrow = TRUE, ncol = 4)
Y <- matrix(c(
  1, 1, 0,
  1, 0, 0,
  1, 0, 0,
  0, 1, 1,
  0, 0, 1
), byrow = TRUE, ncol = 3)
rownames(X) <- c("a1", "a2", "a3", "a4", "a5")
rownames(Y) <- rownames(X)

pareto_dominance(list(set_inclusion(X), set_inclusion(Y)), tau = 2)
```

`partition_centrality` *Partition of centrality by category*

Description

Contribution of the nodes of each category to the centrality of every node (Everett and Borgatti, 2012, 2026).

Usage

```
partition_centrality(
  A,
  B,
  measure = c("betweenness", "degree"),
  digraph = TRUE,
  type = c("out", "in"),
  weighted = FALSE,
  alpha = 1
)
```

Arguments

`A`	A square matrix
`B`	A matrix of the membership of the nodes (rows) in the categories (columns), or a vector with the category of each node
`measure`	The centrality: `betweenness` (default) or `degree`
`digraph`	Whether the matrix is directed or undirected
`type`	For the degree of a directed network, `out` (default) or `in`
`weighted`	Whether the betweenness uses the values of the ties, as in `betweenness_centrality()`
`alpha`	The alpha parameter of Opsahl et al. (2010) for the weighted betweenness

Details

The betweenness of a node v adds up the dependency of every source s on v, $\delta_s(v)$, the extent to which s needs v to reach the other nodes through shortest paths (Brandes, 2001). Grouping the sources by their category, $D^T B$, splits the betweenness of each node into the parts contributed by each category, and with overlapping categories each source contributes in proportion to its memberships. The rows add up to the betweenness of `betweenness_centrality()`.

The degree is split with the alter composition, AB for the out-degree and $A^T B$ for the in-degree.

Value

This function returns a matrix with the part of the centrality of each node (rows) contributed by each category (columns).

Author(s)

Alejandro Espinosa-Rada

References

Brandes, U. (2001). A faster algorithm for betweenness centrality. Journal of Mathematical Sociology, 25(2), 163–177. doi:10.1080/0022250X.2001.9990249

Everett, M. G. and Borgatti, S. P. (2012). Categorical attribute based centrality: E–I and G–F centrality. Social Networks, 34(4), 562–569. doi:10.1016/j.socnet.2012.06.002

Everett, M. G. and Borgatti, S. P. (2026). Alter composition with overlapping group memberships. Social Networks, 85, 80–88. doi:10.1016/j.socnet.2025.12.001

Examples

```
data(campnet)
# Betweenness of the Camp 92 network split by the gender of the sources
partition_centrality(campnet$network, campnet$attributes$gender)
```

---

percolation_clique *Clique percolation*

Description

Clique Percolation Method (CPM) is an algorithm for finding overlapping communities within networks, introduced by Palla et al. (2005). This function firstly identify cliques of size k, then creates a incidence matrix as an affiliation network.

Usage

```
percolation_clique(A)
```

Arguments

A	A matrix

Value

A matrix that assign each node to a clique

Author(s)

Alejandro Espinosa-Rada

References

Palla, G., Derényi, I., Farkas, I., & Vicsek, T. (2005). Uncovering the overlapping community structure of complex networks in nature and society. Nature, 435(7043), 814-818.

Examples

```
A <- matrix(
  c(
    0, 1, 1, 1, 0, 0, 0, 0, 0,
    1, 0, 1, 0, 0, 0, 0, 0, 0,
    1, 1, 0, 1, 0, 0, 0, 0, 0,
    1, 0, 1, 0, 1, 1, 0, 0, 0,
    0, 0, 0, 1, 0, 1, 1, 1, 0,
    0, 0, 0, 1, 1, 0, 1, 1, 0,
    0, 0, 0, 0, 1, 1, 0, 1, 1,
    0, 0, 0, 0, 1, 1, 1, 0, 0,
    0, 0, 0, 0, 0, 0, 1, 0, 0
  ),
  byrow = TRUE, ncol = 9
)
rownames(A) <- letters[1:nrow(A)]
colnames(A) <- letters[1:ncol(A)]
percolation_clique(A)
```

---

perm_label *Permute labels of a matrix*

Description

This function permutes the labels of a matrix.

Usage

```
perm_label(A, m = 1, unique = FALSE)
```

Arguments

A	A matrix
m	Number of permutations
unique	Whether to return unique cases

Value

This function returns the permutation of labels.

Author(s)

Alejandro Espinosa-Rada

Examples

```
W <- matrix(c(
  0, 1, 0, 0, 0,
  0, 0, 1, 0, 0,
  1, 0, 0, 0, 0,
  0, 0, 0, 0, 1,
  0, 0, 0, 1, 0
), byrow = TRUE, ncol = 5)
rownames(W) <- c("P", "Q", "R", "S", "T")
colnames(W) <- rownames(W)
perm_label(W, m = 1000, unique = TRUE)
```

`perm_matrix` *Permutation matrix*

Description

This function create permutation matrices.

Usage

```
perm_matrix(n, m = 1, unique = FALSE)
```

Arguments

`n`	The size of the square matrix
`m`	Number of permutations
`unique`	Whether to return unique cases

Value

This function returns a list of permutation matrices

Author(s)

Alejandro Espinosa-Rada

Examples

```
W <- matrix(c(
  0, 1, 0, 0, 0,
  0, 0, 1, 0, 0,
  1, 0, 0, 0, 0,
  0, 0, 0, 0, 1,
  0, 0, 0, 1, 0
), byrow = TRUE, ncol = 5)
rownames(W) <- c("P", "Q", "R", "S", "T")
```

```
colnames(W) <- rownames(W)
perm_matrix(5, m = 1000, unique = TRUE)
```

---

pos_dominance *Positional dominance on indirect relations*

Description

Dominance between the rows of a matrix of relations, which do not need to be the ties themselves (Brandes, 2016; Schoch and Brandes, 2016).

Usage

```
pos_dominance(
  R,
  map = FALSE,
  benefit = TRUE,
  direction = c("dominated", "dominates")
)
```

Arguments

R	A square matrix of relations, such as the output of `indirect_rel()`
map	Whether the values are sorted before being compared (total homogeneity)
benefit	Whether a larger value is better
direction	Whether `P[u, v] = 1` means that u is `dominated` by v (default) or that u `dominates` v

Details

Neighbourhood inclusion compares the ties of the nodes. The same comparison can be made on any relation derived from the network, such as the distances between the nodes or the number of walks that join them, which is what makes different centrality indices comparable.

Under total heterogeneity (`map = FALSE`) the values are compared one by one: i is dominated by j when its relation with every other node is at most as large. Under total homogeneity (`map = TRUE`) the values are sorted before being compared, so it does not matter with whom the relation is held, only how large the values are.

With `benefit = FALSE` a smaller value is better, which is the case of distances.

Value

This function returns a binary matrix `P` of the positional dominance, where `P[u, v] = 1` when every indirect relation of u is at most the one of v, in the `direction` asked for.

Author(s)

Alejandro Espinosa-Rada

References

Brandes, U. (2016). Network positions. Methodological Innovations, 9, 1–19. doi:10.1177/2059799116630650

Schoch, D. and Brandes, U. (2016). Re-conceptualizing centrality in social networks. European Journal of Applied Mathematics, 27(6), 971–985. doi:10.1017/S0956792516000401

Examples

```
A <- matrix(c(
  0, 1, 1, 1, 0,
  1, 0, 0, 0, 0,
  1, 0, 0, 0, 1,
  1, 0, 0, 0, 0,
  0, 0, 1, 0, 0
), byrow = TRUE, ncol = 5)
rownames(A) <- letters[1:nrow(A)]
colnames(A) <- rownames(A)

# Distances: the closer to the others, the better
D <- indirect_rel(A, type = "distance", digraph = FALSE)
pos_dominance(D, benefit = FALSE)
pos_dominance(D, benefit = FALSE, map = TRUE)
```

---

posneg_index *Positive-negative centrality*

Description

Positive-negative centrality

Usage

```
posneg_index(A, select = c("all", "in", "out"))
```

Arguments

A	A signed symmetric matrix (i.e., with ties that are either -1, 0 or 1)
select	Whether to consider the direction of the outgoing ties. Considering all (default), in or out ties.

Value

This function return the positive-negative centrality index for signed networks (Everett and Borgatti).

Source

Adapted from David Schoch 'signnet'

References

Everett, Martin and Borgatti, Stephen (2014). Networks containing negative ties. Social Networks, 38, 111-120. doi:10.1016/j.socnet.2014.03.005

Examples

```
A <- matrix(
  c(
    0, 1, -1, -1, -1, -1, 0, 0, 0, 0, 0, -1, 0, 0, 1, 1,
    1, 0, -1, 0, -1, -1, 0, 0, -1, -1, 0, 0, 0, 0, 1, 1,
    -1, -1, 0, 1, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0,
    -1, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0,
    -1, -1, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 1, -1, -1,
    -1, -1, 1, 0, 0, 0, 1, 1, -1, 0, 1, 1, -1, 0, 0, -1,
    0, 0, 1, 0, 1, 1, 0, 1, 0, 0, 1, 1, 1, 0, 0, 0,
    0, 0, 1, 1, 0, 1, 1, 0, 0, 0, 1, 1, 0, -1, 0, 0,
    0, -1, 0, 0, 1, -1, 0, 0, 0, 1, -1, 0, 1, 0, -1, 0,
    0, -1, 0, 0, 0, 0, 0, 0, 1, 0, -1, 0, 1, 0, -1, 0,
    0, 0, 0, 0, 0, 1, 1, 1, -1, -1, 0, 1, -1, 0, -1, -1,
    -1, 0, 0, 0, 0, 1, 1, 1, 0, 0, 1, 0, 0, -1, -1, -1,
    0, 0, 0, 0, 0, -1, 1, 0, 1, 1, -1, 0, 0, 1, -1, -1,
    0, 0, 0, 0, 1, 0, 0, -1, 0, 0, 0, -1, 1, 0, 0, -1,
    1, 1, 0, 0, -1, 0, 0, 0, -1, -1, -1, -1, -1, 0, 0, 1,
    1, 1, 0, 0, -1, -1, 0, 0, 0, 0, -1, -1, -1, -1, 1, 0
  ),
  ncol = 16, nrow = 16, byrow = TRUE
)
label <- c(
  "Gavev", "Kotun", "Ove", "Alika", "Nagam", "Gahuk", "Masil", "Ukudz",
  "Notoh", "Kohik", "Geham", "Asaro", "Uheto", "Seuve", "Nagad", "Gama"
)
rownames(A) <- label
colnames(A) <- rownames(A)
posneg_index(A, select = c("all"))
```

---

power_function *Power matrix*

Description

Power of a matrix computed by successive matrix multiplication.

Usage

```
power_function(A, n)
```

Arguments

A	A matrix
n	Positive integer

Value

This function return the power of a matrix by repeating matrix multiplication.

Author(s)

Alejandro Espinosa-Rada

References

Wasserman, S. and Faust, K. (1994). Social network analysis: Methods and applications. Cambridge University Press.

Examples

```
A <- matrix(c(
  1, 0, 0, 0,
  1, 1, 0, 0,
  1, 0, 1, 0,
  0, 1, 1, 1
), byrow = TRUE, ncol = 4, nrow = 4)
power_function(A, 1000)
```

---

pref_attachment *Preferential attachment network*

Description

Random network of Barabasi and Albert (1999), in which the nodes that already have more ties are more likely to receive the ties of the nodes that arrive.

Usage

```
pref_attachment(n, m = 1, power = 1, digraph = FALSE)
```

Arguments

n	The number of nodes
m	The number of ties that every new node creates
power	The power of the degree in the probability of being chosen
digraph	Whether the ties of the new nodes are directed towards the nodes that are already there

Details

The network starts with `m` nodes without ties. Every new node creates `m` ties with the nodes that are already there, choosing each of them with a probability proportional to their degree raised to `power`. With `power = 0` the nodes are chosen at random, and the higher the power, the more the ties concentrate in a few nodes.

Value

This function returns a matrix.

Author(s)

Alejandro Espinosa-Rada

References

Barabasi, A. L. and Albert, R. (1999). Emergence of scaling in random networks. Science, 286(5439), 509–512. doi:10.1126/science.286.5439.509

Examples

```
set.seed(18051889)
A <- pref_attachment(20, m = 2)
gen_degree(A, digraph = FALSE)
```

`preserved_order` *Preserved order*

Description

Whether a centrality index preserves a dominance relation.

Usage

```
preserved_order(
  P,
  scores,
  tol = sqrt(.Machine$double.eps),
  direction = c("dominated", "dominates")
)
```

Arguments

`P`	A binary dominance matrix
`scores`	A vector of centrality scores in the same order as the rows of `P`
`tol`	Numerical tolerance
`direction`	Whether `P[u, v] = 1` means that `u` is `dominated` by `v` (default) or that `u` `dominates` `v`

Details

A centrality index preserves the dominance if no dominated node has a higher score than the node that dominates it (Schoch and Brandes, 2016). Differences smaller than `tol` are considered ties, as eigenvectors and other iterative scores are only approximated.

Value

This function returns whether the order is preserved and the pairs that violate it.

Author(s)

Alejandro Espinosa-Rada

References

Schoch, D. and Brandes, U. (2016). Re-conceptualizing centrality in social networks. European Journal of Applied Mathematics, 27(6), 971–985. doi:10.1017/S0956792516000401

Examples

```
A <- matrix(c(
  0, 1, 1, 1, 0,
  1, 0, 0, 0, 0,
  1, 0, 0, 0, 1,
  1, 0, 0, 0, 0,
  0, 0, 1, 0, 0
), byrow = TRUE, ncol = 5)
rownames(A) <- letters[1:nrow(A)]
colnames(A) <- rownames(A)

P <- neigh_inclusion(A)
preserved_order(P, rowSums(A))
preserved_order(P, betweenness_centrality(A, digraph = FALSE))
```

---

q_analysis *Q-analysis*

Description

Q-analysis of a simplicial complex (Atkin, 1974): the q-connected components at every dimension, the structure vectors, the obstruction vector and the eccentricity of each simplex.

Usage

```
q_analysis(
  A,
  simplicial_complex = FALSE,
  complex = c("clique", "neighbourhood"),
```

```
  closed = FALSE,
  eccentricity = c("atkin", "johnson"),
  dimensions = FALSE
)
```

Arguments

A	An incidence matrix of simplices (rows) and vertices (columns), or a square matrix of a network
simplicial_complex	Whether A is an incidence matrix of simplices (TRUE) or a network (FALSE)
complex	The complex built from a network: the maximal cliques (clique, default) or the neighbourhoods (neighbourhood)
closed	Whether the neighbourhoods include the node itself, for complex = "neighbourhood"
eccentricity	The definition of the eccentricity: atkin (default) or johnson
dimensions	Kept for compatibility with version 1.0-3. The table of the dimensions is always returned

Details

A simplex is a set of vertices, and its dimension q is the number of its vertices minus one. Two simplices are q-near when they share at least $q + 1$ vertices, that is, a face of dimension q, and q-connected when a chain of q-near simplices joins them. For every q from the largest dimension down to 0, the simplices of dimension q or more are grouped in q-connected components (Atkin, 1974; Freeman, 1980). The table of the results has, for each q:

Q, the first structure vector: the number of q-connected components.

n, the second structure vector: the number of simplices of dimension q or more.

Qbar, the third structure vector: $1 - Q/n$, which is zero when no simplex is q-connected to another and approaches one when they all form a single component (Raj et al., 2024).

obstruction, the obstruction vector: $Q - 1$, the number of gaps that separate the components (Atkin, 1974).

The eccentricity measures how much a simplex stands apart from the others, and there are two definitions. With eccentricity = "atkin" (default) it is $(\hat{q} - \check{q})/(\check{q} + 1)$, where $\hat{q}$ is the dimension of the simplex and $\check{q}$ the dimension of the largest face it shares with another simplex (Atkin, 1974). It is zero for a simplex that is a face of another, and infinite for a simplex that shares no vertex with the others. With eccentricity = "johnson" it is the family eccentricity of Johnson, the smallest proportion of the vertices of the simplex that are not in another simplex, $\min_{\sigma'} |\sigma \setminus \sigma'|/|\sigma|$, as implemented by Smirnov et al. (2025). It runs from zero to one, which makes simplices of different dimension comparable, and it is NA when the complex has a single simplex.

With simplicial_complex = TRUE, the rows of A are the simplices and the columns their vertices, as in the example of Freeman (1980), where the researchers are simplices of the events that linked them. The conjugate complex, in which the columns are the simplices, is the analysis of t(A).

With simplicial_complex = FALSE, A is a network and the complex is built from it (Raj et al., 2024): with complex = "clique", the simplices are the maximal cliques of the underlying undirected network, including the isolated nodes as simplices of dimension 0; with complex = "neighbourhood",

each node is the simplex of its neighbours, the rows of A (the out-neighbours of a directed network). With closed = TRUE the node is also a vertex of its own simplex (closed neighbourhood), so that two adjacent nodes share at least the two of them. The rows without vertices are not simplices. The complex is the one returned by simplicial_complexes().

Value

This function returns a list with the incidence matrix of the simplices analysed, the q_table with the structure and obstruction vectors, the components at each q (named q3, q2, ...), and the eccentricity of each simplex.

Author(s)

Alejandro Espinosa-Rada

References

Atkin, R. H. (1974). Mathematical structure in human affairs. New York: Crane, Rusak.

Freeman, L. C. (1980). Q-analysis and the structure of friendship networks. International Journal of Man-Machine Studies, 12(4), 367–378. doi:10.1016/S00207373(80)800216

Raj, U., Banerjee, A., Ray, S. and Bhattacharya, S. (2024). Structure of higher-order interactions in social-ecological networks through Q-analysis of their neighbourhood and clique complex. PLOS ONE, 19(8), e0306409. doi:10.1371/journal.pone.0306409

Smirnov, N., Kurkin, S. and Hramov, A. E. (2025). A Q-analysis package for higher-order interactions analysis in Python and its application in network physiology. Frontiers in Network Physiology, 5. doi:10.3389/fnetp.2025.1691159

Examples

```
# Freeman (1980): 29 researchers (simplices) and the 19 events that linked them (vertices)
A <- matrix(c(
  0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
  0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0,
  0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 1, 0, 0, 0,
  0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
  0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0,
  0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0,
  0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
  0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0,
  0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
  0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0,
  0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1,
  0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0,
  0, 1, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1,
  0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
  1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
  1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
  0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0,
  0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0,
  0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0,
```

```
  0, 1, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
  0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0,
  0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
  0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0,
  0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0,
  0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
  0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
  0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
  0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0,
  0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0
), byrow = TRUE, ncol = 19)
colnames(A) <- letters[1:ncol(A)]
rownames(A) <- 1:nrow(A)

Q <- q_analysis(A, simplicial_complex = TRUE)
Q$q_table
Q$components$q3

# A network: a clique of four nodes and a pendant node
B <- matrix(c(
  0, 1, 1, 1, 0,
  1, 0, 1, 1, 0,
  1, 1, 0, 1, 0,
  1, 1, 1, 0, 1,
  0, 0, 0, 1, 0
), byrow = TRUE, ncol = 5)
rownames(B) <- letters[1:nrow(B)]
colnames(B) <- rownames(B)

q_analysis(B, complex = "clique")$q_table
q_analysis(B, complex = "neighbourhood")$q_table
```

---

qap_cor *QAP correlation*

Description

Correlation between two matrices, with a test based on the permutation of the nodes (Hubert and Schultz, 1976; Krackhardt, 1987).

Usage

```
qap_cor(
  A,
  B,
  reps = 1000,
  diag = FALSE,
  method = c("pearson", "spearman", "kendall")
)
```

Arguments

A	A square matrix
B	A square matrix of the same order
reps	Number of permutations
diag	Whether the diagonal is considered
method	Correlation coefficient: pearson (default), spearman or kendall

Details

The ties of a network are not independent, so the usual test of a correlation does not apply. The quadratic assignment procedure compares the observed correlation with the correlations obtained after permuting the rows and the columns of one of the matrices at the same time, which keeps its structure while breaking its association with the other matrix.

Value

This function returns the observed correlation and the proportion of permutations with a correlation greater or equal, lower or equal, and larger in absolute value, than the observed one.

Author(s)

Alejandro Espinosa-Rada

References

Hubert, L. and Schultz, J. (1976). Quadratic assignment as a general data analysis strategy. British Journal of Mathematical and Statistical Psychology, 29(2), 190–241. doi:10.1111/j.20448317.1976.tb00714.x

Krackhardt, D. (1987). QAP partialling as a test of spuriousness. Social Networks, 9(2), 171–186. doi:10.1016/03788733(87)900128

Examples

```
A <- matrix(c(
  0, 1, 1, 0,
  1, 0, 1, 0,
  1, 1, 0, 1,
  0, 0, 1, 0
), byrow = TRUE, ncol = 4)
B <- matrix(c(
  0, 1, 0, 0,
  1, 0, 1, 0,
  0, 1, 0, 1,
  0, 0, 1, 0
), byrow = TRUE, ncol = 4)

set.seed(18051889)
qap_cor(A, B, reps = 100)
```

qap_lm *QAP regression*

Description

Regression between matrices, with a test based on the permutation of the nodes (Krackhardt, 1988; Dekker, Krackhardt and Snijders, 2007).

Usage

```
qap_lm(
  Y,
  X,
  reps = 1000,
  family = c("gaussian", "binomial"),
  method = c("dsp", "y"),
  diag = FALSE
)
```

Arguments

Y	A square matrix with the dependent relation
X	A list of square matrices with the independent relations
reps	Number of permutations
family	gaussian for a linear regression (default) or binomial for a logistic regression
method	Whether to permute the dependent matrix (y) or the residuals of each predictor (dsp, default)
diag	Whether the diagonal is considered

Details

The coefficients are those of an ordinary regression (or a logistic regression when family = "binomial") of the ties of Y on the ties of the matrices in X. As the ties are not independent, the standard errors of the regression do not apply, and the coefficients are compared with the ones obtained after permuting the nodes.

With method = "y" the rows and columns of Y are permuted. With method = "dsp" (default) the double semi-partialling of Dekker et al. (2007) is used: each predictor is regressed on the other predictors, and the residuals of that regression are permuted, which behaves better when the predictors are correlated with each other. The intercept is not permuted by the double semi-partialling, so its p-values are not returned.

Value

This function returns the coefficients, the proportion of permutations with a coefficient greater or equal, lower or equal, and larger in absolute value, than the observed one, and the fit of the model.

Author(s)

Alejandro Espinosa-Rada

References

Dekker, D., Krackhardt, D. and Snijders, T. A. B. (2007). Sensitivity of MRQAP tests to collinearity and autocorrelation conditions. Psychometrika, 72(4), 563–581. doi:10.1007/s1133600790161

Krackhardt, D. (1988). Predicting with networks: Nonparametric multiple regression analysis of dyadic data. Social Networks, 10(4), 359–381. doi:10.1016/03788733(88)900044

Examples

```
set.seed(18051889)
Y <- matrix(c(
  0, 1, 1, 0, 0,
  1, 0, 1, 0, 0,
  1, 1, 0, 1, 0,
  0, 0, 1, 0, 1,
  0, 0, 0, 1, 0
), byrow = TRUE, ncol = 5)
X1 <- matrix(c(
  0, 1, 0, 0, 0,
  1, 0, 1, 0, 0,
  0, 1, 0, 1, 0,
  0, 0, 1, 0, 1,
  0, 0, 0, 1, 0
), byrow = TRUE, ncol = 5)

qap_lm(Y, list(distance = X1), reps = 100)
```

---

recip_coef *Reciprocity*

Description

This measure calculated the reciprocity of an asymmetric matrix (directed graph).

Usage

```
recip_coef(
  A,
  diag = NULL,
  method = c("total_ratio", "ratio_nonnull", "global")
)
```

Arguments

A	A matrix
diag	Whether to consider the diagonal of the matrix
method	Whether to use total_ratio, ratio_nonnull or global reciprocity

Value

Return a reciprocity coefficient

Author(s)

Alejandro Espinosa-Rada

References

Wasserman, S. and Faust, K. (1994). Social network analysis: Methods and applications. Cambridge University Press.

Examples

```
A <- matrix(c(0, 1, 1, 0,
              1, 0, 1, 0,
              0, 0, 0, 0,
              1, 0, 0, 0), byrow = TRUE, ncol = 4)
recip_coef(A)
```

---

redundancy *Redundancy measures*

Description

Redundancy measures of the structural holes theory for binary matrixes

Usage

```
redundancy(A, ego = NULL, digraph = FALSE, weighted = FALSE)
```

Arguments

A	A symmetric matrix object
ego	Name of ego in the matrix
digraph	Whether the matrix is directed or undirected
weighted	Whether the matrix is weighted or not

Value

This function returns redundancy, effective size and efficiency measures (Burt, 1992).

Author(s)

Alejandro Espinosa-Rada

References

Burt, R.S., 1992. Structural Holes: the Social Structure of Competition. Harvard University Press, Cambridge.

Borgatti, S., 1997. Unpacking Burt's redundancy measure. Connections, 20(1): 35-38.

Examples

```
A <- matrix(c(
  0, 1, 0, 0, 1, 1, 1,
  1, 0, 0, 1, 0, 0, 1,
  0, 0, 0, 0, 0, 0, 1,
  0, 1, 0, 0, 0, 0, 1,
  1, 0, 0, 0, 0, 0, 1,
  1, 0, 0, 0, 0, 0, 1,
  1, 1, 1, 1, 1, 1, 0
), ncol = 7, byrow = TRUE)
rownames(A) <- letters[1:nrow(A)]
colnames(A) <- letters[1:ncol(A)]
redundancy(A, ego = "g")
```

---

rege *Regular equivalence*

Description

REGE algorithm (White and Reitz, 1983; Borgatti and Everett, 1993): two nodes are regularly equivalent when they are connected to nodes that are themselves equivalent, even if they are not connected to the same nodes.

Usage

```
rege(A, iter = 3)
```

Arguments

A	A square matrix, which can be valued
iter	Number of iterations

Details

Structural equivalence asks for the same neighbours, while regular equivalence only asks for neighbours that play the same role. Each alter of a node is matched with the alter of the other node that resembles it the most, and the similarities are computed again with the matches of the previous iteration.

Value

This function returns a matrix with the regular equivalence of every pair of nodes, between zero and one.

Author(s)

Alejandro Espinosa-Rada

References

Borgatti, S. P. and Everett, M. G. (1993). Two algorithms for computing regular equivalence. Social Networks, 15(4), 361–376. doi:10.1016/03788733(93)90012A

White, D. R. and Reitz, K. P. (1983). Graph and semigroup homomorphisms on networks of relations. Social Networks, 5(2), 193–234. doi:10.1016/03788733(83)900254

Examples

```
# Two managers with different subordinates play the same role
A <- matrix(c(
  0, 1, 1, 0, 0, 0, 0,
  0, 0, 0, 1, 1, 0, 0,
  0, 0, 0, 0, 0, 1, 1,
  0, 0, 0, 0, 0, 0, 0,
  0, 0, 0, 0, 0, 0, 0,
  0, 0, 0, 0, 0, 0, 0,
  0, 0, 0, 0, 0, 0, 0
), byrow = TRUE, ncol = 7)
rownames(A) <- letters[1:nrow(A)]
colnames(A) <- rownames(A)

round(rege(A), 3)
```

---

segregation *Segregation measures*

Description

Measures of how much the ties of a network stay within the groups given by an attribute of the nodes, reviewed by Bojanowski and Corten (2014).

Usage

```
segregation(
  A,
  att,
  method = c("assortativity", "gam", "orwg", "coleman", "freeman"),
  digraph = FALSE,
  loops = FALSE
)
```

Arguments

A	A square matrix
att	A vector with the group of each node
method	The measure: assortativity (default), gam, orwg, coleman or freeman
digraph	Whether the matrix is directed or undirected. The coleman index is defined for directed networks
loops	Whether to consider the loops of the matrix

Details

All the measures are computed from the mixing matrix, which counts the ties within and between groups:

assortativity: the proportion of ties that are within groups, compared with the proportion expected if the ties were distributed at random keeping how active each group is (Newman, 2003). It is one when every tie is within a group, and zero under random mixing.

gam: the index of Gupta, Anderson and May (1989), the trace of the matrix of the proportion of the ties of each group that go to every other group, rescaled to go from $-1/(K-1)$ to one. It is defined for undirected networks, and every group should have at least one tie.

orwg: the odds of a tie within a group divided by the odds of a tie between groups (Moody, 2001). Unlike the other measures, it takes into account the pairs of nodes that are not tied, so it is not affected by the density of the network.

coleman: the homophily index of Coleman (1958), computed for each group: how many ties the group sends to itself compared with the ties it would send if it chose the other nodes at random. It is one when the group only relates to itself.

freeman: the segregation index of Freeman (1978) for two groups: how many fewer ties between the groups there are than the ones expected in a random network with the same density and group sizes. It is zero when there are as many as expected, or more.

Value

This function returns the value of the measure, which is a value per group for the coleman index.

Author(s)

Alejandro Espinosa-Rada

References

Bojanowski, M. and Corten, R. (2014). Measuring segregation in social networks. Social Networks, 39, 14–32. doi:10.1016/j.socnet.2014.04.001

Coleman, J. (1958). Relational analysis: The study of social organizations with survey methods. Human Organization, 17(4), 28–36. doi:10.17730/humo.17.4.q5604m676260q8n7

Freeman, L. C. (1978). Segregation in social networks. Sociological Methods and Research, 6(4), 411–429. doi:10.1177/004912417800600401

Gupta, S., Anderson, R. M. and May, R. M. (1989). Networks of sexual contacts: implications for the pattern of spread of HIV. AIDS, 3(12), 807–817. doi:10.1097/00002030-198912000-00005

Moody, J. (2001). Race, school integration, and friendship segregation in America. American Journal of Sociology, 107(3), 679–716. doi:10.1086/338954

Newman, M. E. J. (2003). Mixing patterns in networks. Physical Review E, 67(2), 026126. doi:10.1103/PhysRevE.67.026126

Examples

```
A <- matrix(c(
  0, 1, 1, 0, 0, 0,
  1, 0, 1, 0, 0, 0,
  1, 1, 0, 1, 0, 0,
  0, 0, 1, 0, 1, 1,
  0, 0, 0, 1, 0, 1,
  0, 0, 0, 1, 1, 0
), byrow = TRUE, ncol = 6)
rownames(A) <- letters[1:nrow(A)]
colnames(A) <- rownames(A)
att <- c("a", "a", "a", "b", "b", "b")

segregation(A, att)
segregation(A, att, method = "orwg")
segregation(A, att, method = "coleman", digraph = FALSE)
```

---

set_inclusion *Set inclusion of neighbourhoods*

Description

Inclusion of the neighbourhoods of the rows of an incidence matrix.

Usage

```
set_inclusion(N, M = N, proper = FALSE)
```

Arguments

N	An incidence matrix of the neighbourhoods to be included
M	An incidence matrix, with the same dimensions as N, of the including neighbourhoods
proper	Whether the inclusion should be proper

Details

The neighbourhoods of the rows can be defined on any set of columns (e.g. papers of authors, or events of actors). `N` contains the neighbourhood that should be included, and `M` the neighbourhood in which it should be included. `M` is the same as `N` for open neighbourhoods, or can add other elements, such as the node itself, for closed neighbourhoods.

With `proper = TRUE` the inclusion should be proper, $N(u) \subsetneq M(v)$, i.e. $M(v)$ has at least one element that is not in $N(u)$.

Value

This function returns a binary matrix `P` where `P[u, v] = 1` if `N(u)` is included in `M(v)`.

Author(s)

Alejandro Espinosa-Rada

Examples

```
X <- matrix(c(
  1, 1, 1, 0,
  1, 1, 0, 0,
  1, 0, 0, 0,
  0, 1, 0, 0,
  0, 0, 1, 1
), byrow = TRUE, ncol = 4)
rownames(X) <- c("a1", "a2", "a3", "a4", "a5")
colnames(X) <- c("w1", "w2", "w3", "w4")

set_inclusion(X)
```

---

shared_partners *Shared partners*

Description

Shared partners

Usage

```
shared_partners(
  A,
  loops = FALSE,
  directed = TRUE,
  type = c("dsp", "esp", "nsp")
)
```

Arguments

A	A binary matrix
loops	Whether to consider the loops
directed	Whether the matrix is directed
type	Whether to return the dyad-wise (dsp) (default), edge-wise (esp) or non-edgewise (nsp) shared partners (Hunter and Handcock, 2006)

Value

This function return the distribution of shared partners.

Author(s)

Alejandro Espinosa-Rada

References

Hunter, D. R. and M. S. Handcock (2006), Inference in curved exponential family models for networks, Journal of Computational and Graphical Statistics, 15: 565– 583.

Examples

```
A <- matrix(c(
  0, 1, 0, 0, 0, 0,
  1, 0, 1, 1, 0, 1,
  0, 1, 0, 1, 0, 0,
  0, 1, 1, 0, 1, 1,
  0, 0, 0, 1, 0, 1,
  0, 1, 0, 1, 1, 0
), byrow = TRUE, ncol = 6)
shared_partners(A, type = "dsp")
shared_partners(A, type = "esp")
shared_partners(A, type = "nsp")
```

---

simplicial_complexes *Simplicial complexes*

Description

Incidence matrix of the nodes of a network and the simplices of its clique complex or of its neighbourhood complex.

Usage

```
simplicial_complexes(
  A,
  zero_simplex = TRUE,
  projection = FALSE,
  complex = c("clique", "neighbourhood"),
  closed = FALSE,
  valued = FALSE
)
```

Arguments

A	A square matrix of a network, with names.
zero_simplex	Whether to include the isolated nodes as simplices of dimension 0, for complex = "clique".
projection	Whether to return the links between the simplices through their shared nodes, and between the nodes through their shared simplices.
complex	The complex: the maximal cliques (clique, default) or the neighbourhoods (neighbourhood).
closed	Whether the neighbourhoods include the node itself, for complex = "neighbourhood".
valued	Whether the projections count the shared nodes or simplices instead of indicating whether there are any.

Details

A simplex is a set of nodes, and a simplicial complex a collection of simplices that contains every face of its simplices (Atkin, 1974). The complex is represented by its maximal simplices, as the faces are implied by the simplices that contain them.

With complex = "clique" (default), the simplices are the maximal cliques of the underlying undirected network, so that a clique of four nodes is a single simplex of dimension 3. The isolated nodes are cliques of a single node, and they are included as simplices of dimension 0 when zero_simplex = TRUE.

With complex = "neighbourhood", each node is the simplex of its neighbours, the rows of A (the out-neighbours of a directed network), and with closed = TRUE the node is also a vertex of its own simplex (Raj et al., 2024). The nodes without neighbours do not form a simplex unless the neighbourhoods are closed.

The rows of the result are the nodes and the columns the simplices, named after their nodes (a-b-c) for the clique complex, and after the node whose neighbourhood they are (N(a) or N[a]) for the neighbourhood complex. q_analysis(t(S), simplicial_complex = TRUE) is the Q-analysis of the simplices, and q_analysis(S, simplicial_complex = TRUE) that of the conjugate complex, in which the nodes are connected through the simplices they share.

Value

This function returns the incidence matrix of the nodes (rows) and the simplices (columns). With projection = TRUE, a list with the incidence matrix (simplex), the projection of the simplices (projection1) and the projection of the nodes (projection2).

Author(s)

Alejandro Espinosa-Rada

References

Atkin, R. H. (1974). Mathematical structure in human affairs. New York: Crane, Rusak.

Freeman, L. C. (1980). Q-analysis and the structure of friendship networks. International Journal of Man-Machine Studies, 12(4), 367–378. doi:10.1016/S00207373(80)800216

Raj, U., Banerjee, A., Ray, S. and Bhattacharya, S. (2024). Structure of higher-order interactions in social-ecological networks through Q-analysis of their neighbourhood and clique complex. PLOS ONE, 19(8), e0306409. doi:10.1371/journal.pone.0306409

Wasserman, S. and Faust, K. (1994). Social network analysis: Methods and applications. Cambridge University Press.

Examples

```
A <- matrix(c(
  0, 1, 1, 0, 0, 0, 0, 1, 0,
  1, 0, 1, 0, 0, 0, 0, 0, 0,
  1, 1, 0, 0, 0, 0, 0, 0, 0,
  0, 0, 0, 0, 1, 1, 0, 0, 0,
  0, 0, 0, 1, 0, 0, 0, 0, 0,
  0, 0, 0, 1, 0, 0, 1, 1, 0,
  0, 0, 0, 0, 0, 1, 0, 1, 0,
  1, 0, 0, 0, 0, 1, 1, 0, 0,
  0, 0, 0, 0, 0, 0, 0, 0, 0
), byrow = TRUE, ncol = 9)
rownames(A) <- letters[1:nrow(A)]
colnames(A) <- rownames(A)

simplicial_complexes(A)
simplicial_complexes(A, zero_simplex = FALSE)
simplicial_complexes(A, complex = "neighbourhood", closed = TRUE)
simplicial_complexes(A, projection = TRUE, valued = TRUE)$projection2
```

---

small_world *Small world network*

Description

Random network of Watts and Strogatz (1998), which has the short distances of a random network and the high transitivity of a regular one.

Usage

```
small_world(n, neighbours = 2, p = 0.05)
```

Arguments

n	The number of nodes
neighbours	The number of closest nodes on each side that every node is tied to in the ring
p	Probability of rewiring each tie

Details

The network starts as a ring in which every node is tied to its neighbours closest nodes on each side. Then each tie is rewired with probability p: one of its ends is replaced by a node chosen at random, avoiding loops and repeated ties. With p = 0 the ring is left as it is, and with p = 1 every tie is rewired.

Value

This function returns a symmetric matrix.

Author(s)

Alejandro Espinosa-Rada

References

Watts, D. J. and Strogatz, S. H. (1998). Collective dynamics of 'small-world' networks. Nature, 393(6684), 440–442. doi:10.1038/30918

Examples

```
set.seed(18051889)
A <- small_world(20, neighbours = 2, p = 0.05)
gen_degree(A, digraph = FALSE)
```

---

social_influence *Social influence*

Description

Simulates how the opinions of the actors change when they are influenced by their neighbours, under the rules compared by Flache et al. (2017).

Usage

```
social_influence(
  A,
  opinion,
  rule = c("assimilation", "bounded", "repulsion", "friedkin"),
  mu = 0.3,
  epsilon = 0.15,
```

```
  susceptibility = 0.5,
  steps = 50
)
```

Arguments

`A`	A square matrix of influence, which can be weighted. The rows are usually normalised so that the weights of each actor add up to one
`opinion`	A vector with the initial opinion of every actor, usually between zero and one
`rule`	The rule of influence: `assimilation` (default), `bounded`, `repulsion` or `friedkin`
`mu`	How much an actor moves at each step
`epsilon`	Maximum difference of opinion that still influences an actor, for the `bounded` rule
`susceptibility`	Weight given to the neighbours in the `friedkin` rule, as a number or a vector with one value per actor
`steps`	Number of steps

Details

Every actor starts with an opinion and updates it at each step:

`assimilation`: the actor moves towards the opinions of its neighbours, which leads the network to consensus (French, 1956; DeGroot, 1974).

`bounded`: the actor is only influenced by the neighbours whose opinion differs less than `epsilon`, which leads to fragmentation into groups that no longer influence each other (Hegselmann and Krause, 2002).

`repulsion`: the actor moves towards similar neighbours and away from those who are too different, which leads to bi-polarization. The opinions are kept between zero and one.

`friedkin`: the actor combines the opinions of its neighbours with the opinion it started with, and `susceptibility` is the weight given to the neighbours (Friedkin and Johnsen, 1990).

Value

This function returns the opinions of every actor at every step, the final opinions, and the number of groups of opinions at the end.

Author(s)

Alejandro Espinosa-Rada

References

DeGroot, M. H. (1974). Reaching a consensus. Journal of the American Statistical Association, 69(345), 118–121. doi:10.1080/01621459.1974.10480137

Flache, A., Mas, M., Feliciani, T., Chattoe-Brown, E., Deffuant, G., Huet, S. and Lorenz, J. (2017). Models of social influence: Towards the next frontiers. Journal of Artificial Societies and Social Simulation, 20(4), 2. doi:10.18564/jasss.3521

French, J. R. P. (1956). A formal theory of social power. Psychological Review, 63(3), 181–194. doi:10.1037/h0046123

Friedkin, N. E. and Johnsen, E. C. (1990). Social influence and opinions. Journal of Mathematical Sociology, 15(3-4), 193–206. doi:10.1080/0022250X.1990.9990069

Hegselmann, R. and Krause, U. (2002). Opinion dynamics and bounded confidence models, analysis and simulation. Journal of Artificial Societies and Social Simulation, 5(3), 2.

Examples

```
A <- matrix(c(
  0, 1, 1, 0, 0, 0,
  1, 0, 1, 0, 0, 0,
  1, 1, 0, 1, 0, 0,
  0, 0, 1, 0, 1, 1,
  0, 0, 0, 1, 0, 1,
  0, 0, 0, 1, 1, 0
), byrow = TRUE, ncol = 6)
rownames(A) <- letters[1:nrow(A)]
colnames(A) <- rownames(A)
W <- A / rowSums(A)
opinion <- c(0.1, 0.2, 0.3, 0.7, 0.8, 0.9)

social_influence(W, opinion, rule = "assimilation", steps = 20)$final
social_influence(W, opinion, rule = "bounded", epsilon = 0.15, steps = 20)$groups
```

---

spatial_cor *Spatial autocorrelation*

Description

This function calculate some spatial autocorrelations for a sample of networks at different orders (distances).

Usage

```
spatial_cor(
  A,
  V,
  measures = c("covariance", "correlation", "moran", "geary"),
  mean = TRUE,
  diag = FALSE,
  distance1 = TRUE,
  rowstand = FALSE,
  scale = FALSE
)
```

Arguments

A	A symmetric matrix
V	A vector
measures	Whether to use the Covariance covariance (default), Correlation correlation, Moran I moran or Geary's C geary
mean	Whether to use the mean of the vector for the measures
diag	Whether to consider the diagonal of the matrix for the measures
distance1	Whether to return only the spatial autocorrelation considering the actor at distance 1
rowstand	Whether to use the row-standardization to estimate Moran I (Anselin, 1995)
scale	Whether to scale Moran I (Anselin, 1995)

Value

This function return the global spatial autocorrelation. Multiple orders can also be computed.

References

Anselin, L. (1995). Local indicators of spatial association—LISA. Geographical analysis, 27(2), 93-115.

Geary, R.C. (1954). "The Contiguity Ratio and Statistical Mapping." The Incorporated Statistician, 5: 115-145.

Moran, P.A.P. (1950). "Notes on Continuous Stochastic Phenomena." Biometrika, 37: 17-23.

Examples

```
A <- matrix(c(
  0, 0, 1, 1,
  0, 0, 1, 0,
  1, 0, 0, 0,
  1, 0, 1, 0
), byrow = TRUE, ncol = 4)
V <- c(2, 2, 1, 1)

spatial_cor(A, V, measures = c("moran"))
```

---

struc_balance *Structural balance*

Description

Structural balance

Usage

```
struc_balance(A, B = NULL, score = c("triangle", "walk"))
```

Arguments

A	A signed symmetric matrix (i.e., with ties that are either -1, 0 or 1)
B	A signed symmetric matrix considered as the negative ties (i.e., with ties that are either -1, 0 or 1)
score	Whether to return the `triangle` (default) or `walk` balance score (Aref and Wilson, 2017)

Value

This function return the structural balance (Heider, 1940; Cartwright and Harary, 1956). When `B` is used, matrix A is considered the negative matrix and `A` the positive matrix.

Author(s)

Alejandro Espinosa-Rada

References

Aref, Samin and Wilson, Mark C. (2017). Measuring partial balance in signed networks. Journal of Complex Networks, 6(4): 566-595.

Cartwright, Dorwin, and Harary, Frank (1956). Structural balance: a generalization of Heider's theory. Psychological review, 63(5), 277.

Heider, Fritz (1946). Attitudes and Cognitive Organization. The Journal of Psychology, 21: 107–112

Examples

```
A <- matrix(c(
  0, -1, -1, 0,
  -1, 0, 1, 0,
  -1, 1, 0, 0,
  0, 0, 0, 0
), byrow = TRUE, ncol = 4)
rownames(A) <- letters[1:nrow(A)]
colnames(A) <- rownames(A)
struc_balance(A)
```

`structural_holes` *Structural holes*

Description

Effective size, efficiency and constraint of Burt (1992) for every node of a valued or directed network, with the option of treating the alters of the same category as redundant (Everett and Borgatti, 2026).

Usage

```
structural_holes(A, B = NULL, beta = 1, ego_network = TRUE)
```

Arguments

A	A square matrix, binary or valued
B	An optional matrix of the membership of the nodes (rows) in the categories (columns), or a vector with the category of each node
beta	The strength of the tie added between two alters of the same category
ego_network	Whether the measures are computed within the ego network of each node (TRUE) or within the whole network

Details

Burt (1992) measures the ties of a node i with its alters j through the proportion of the ties of i that go to j, $p_{ij} = (a_{ij} + a_{ji}) / \sum_k (a_{ik} + a_{ki})$, so a directed tie counts in both directions. The effective size is $\sum_j (1 - \sum_q p_{iq} m_{jq})$, where m_{jq} is the tie of j with q divided by the strongest tie of j; the efficiency is the effective size divided by the number of alters; and the constraint is $\sum_j (p_{ij} + \sum_q p_{iq} p_{qj})^2$.

With `ego_network = TRUE` (default, as in UCINET) the proportions are computed within the ego network of each node, and with `ego_network = FALSE` within the whole network, as in `igraph::constraint()`. For binary undirected networks and ego networks, the effective size is the one of `redundancy()` and the constraint the one of `eb_constraint()`.

When `B` is given, two alters of the same category are treated as partly redundant even when they are not tied: each missing tie between two alters is given the value $\beta \sum_k B_{xk} B_{yk}$, the product of their memberships, before the measures are computed (Everett and Borgatti, 2026: Eq. 6). With a partition and `beta = 1` two alters of the same category count as tied; `beta = 0` gives the original measures, and the values in between set how much the category matters.

Value

This function returns a data frame with the number of alters, the effective size, the efficiency and the constraint of each node.

Author(s)

Alejandro Espinosa-Rada

References

Burt, R.S., 1992. Structural Holes: the Social Structure of Competition. Harvard University Press, Cambridge.

Everett, M. G. and Borgatti, S. P. (2026). Alter composition with overlapping group memberships. Social Networks, 85, 80–88. doi:10.1016/j.socnet.2025.12.001

Examples

```
data(campnet)
structural_holes(campnet$network)

# Alters of the same gender are partly redundant
structural_holes(campnet$network, B = campnet$attributes$gender, beta = 0.5)
```

structural_na *Structural Missing Data*

Description

Assign NA to missing data in matrices.

Usage

```
structural_na(
  A,
  label = NULL,
  row_labels = NULL,
  col_labels = NULL,
  two_mode = FALSE
)
```

Arguments

A	An incident or symmetric matrix object.
label	A string vector with the names of the theoretical complete matrix (used for one-mode networks only).
row_labels	A string vector with the names of the rows (used for two-mode networks).
col_labels	A string vector with the names of the columns (used for two-mode networks).
two_mode	Boolean indicating whether the matrix is two-mode. Default is FALSE.

Value

This function returns a matrix with NA assigned to missing data.

Examples

```
# Example for one-mode network
A <- matrix(c(
  0, 1, 1,
  1, 0, 1,
  0, 0, 0
), byrow = TRUE, ncol = 3)
colnames(A) <- c("A", "C", "D")
rownames(A) <- c("A", "C", "D")
label <- c("A", "B", "C", "D", "E")
structural_na(A, label = label)

# Example for two-mode network
B <- matrix(c(
  0, 1, 0,
  1, 0, 1,
```

```
  0, 1, 0,
  1, 0, 1
), byrow = TRUE, ncol = 3)
rownames(B) <- c("X1", "X2", "X3", "X4")
colnames(B) <- c("Y1", "Y2", "Y3")
rlabels <- c("X1", "X2", "X3", "X4", "X5")
clabels <- c("Y1", "Y2", "Y3", "Y4")
structural_na(B, row_labels = rlabels, col_labels = clabels, two_mode = TRUE)
```

`supra_adjacency` *Supra-adjacency matrix*

Description

Arranges the layers of a multiplex network, in which the same actors are connected by several relations, into a single matrix of actor-layer pairs (De Domenico et al., 2013; Kivela et al., 2014).

Usage

```
supra_adjacency(
  layers,
  coupling = c("categorical", "ordinal", "none"),
  weight = 1,
  sparse = FALSE
)
```

Arguments

`layers`	A list of square matrices of the same order, one for each layer, with the same actors in the same order
`coupling`	Whether the copies of an actor are joined in every pair of layers (`categorical`, default), only in consecutive layers (`ordinal`), or not at all (`none`)
`weight`	The value of the ties between the copies of the same actor
`sparse`	Whether to return a sparse matrix of the `Matrix` package

Details

The supra-adjacency matrix has one row and one column for each actor in each layer. The blocks of the diagonal are the layers, and the blocks outside it are the coupling between the layers, which joins the copies of the same actor.

`coupling = "categorical"` (default) joins the copies of an actor in every pair of layers, which is the usual choice when the layers are different relations. `"ordinal"` joins them only in consecutive layers, for layers that follow an order, such as time. `"none"` leaves the layers apart, and the result is block diagonal.

The `weight` of the coupling is the value given to those ties, and it sets how much the layers are held together in measures computed on the whole structure.

The rows and the columns are named after the actor and the layer, as `actor_layer`, taking the names of the matrices and of the list of layers when they have them.

Value

This function returns the supra-adjacency matrix of the layers.

Author(s)

Alejandro Espinosa-Rada

References

De Domenico, M., Sole-Ribalta, A., Cozzo, E., Kivela, M., Moreno, Y., Porter, M. A., Gomez, S. and Arenas, A. (2013). Mathematical formulation of multilayer networks. Physical Review X, 3(4), 041022. doi:10.1103/PhysRevX.3.041022

Kivela, M., Arenas, A., Barthelemy, M., Gleeson, J. P., Moreno, Y. and Porter, M. A. (2014). Multilayer networks. Journal of Complex Networks, 2(3), 203-271. doi:10.1093/comnet/cnu016

Examples

```
A1 <- matrix(c(
  0, 1, 0,
  1, 0, 1,
  0, 1, 0
), byrow = TRUE, ncol = 3, dimnames = list(letters[1:3], letters[1:3]))

A2 <- matrix(c(
  0, 0, 1,
  0, 0, 0,
  1, 0, 0
), byrow = TRUE, ncol = 3, dimnames = list(letters[1:3], letters[1:3]))

supra_adjacency(list(advice = A1, friendship = A2))

# Layers that follow an order are coupled only with the next one
supra_adjacency(list(A1, A2), coupling = "ordinal")
```

---

threshold_diffusion *Threshold diffusion*

Description

Simulates the diffusion of a behaviour through the network, where an actor adopts it when enough of its neighbours have already adopted it (Granovetter, 1978; Valente, 1996).

Usage

```
threshold_diffusion(
  A,
  seeds,
  threshold = 0.5,
```

```
  mode = c("proportion", "count"),
  steps = NULL
)
```

Arguments

A	A square matrix
seeds	The actors that have adopted at the beginning, by name or by position
threshold	The threshold of every actor, as a single value or a vector
mode	Whether the threshold is a proportion of the neighbours (default) or a count of them
steps	Maximum number of steps. If NULL, the process runs until nobody else adopts

Details

At each step, the actors that have not adopted count how many of their neighbours did. They adopt when that number, or that proportion of their neighbours, reaches their threshold. The actors that adopt never go back, so the process stops when nobody else adopts.

Value

This function returns who has adopted at every step, the step in which every actor adopted, and the proportion of actors that adopted.

Author(s)

Alejandro Espinosa-Rada

References

Granovetter, M. (1978). Threshold models of collective behavior. American Journal of Sociology, 83(6), 1420–1443. doi:10.1086/226707

Valente, T. W. (1996). Social network thresholds in the diffusion of innovations. Social Networks, 18(1), 69–89. doi:10.1016/03788733(95)002561

Examples

```
A <- matrix(c(
  0, 1, 1, 0, 0, 0,
  1, 0, 1, 0, 0, 0,
  1, 1, 0, 1, 0, 0,
  0, 0, 1, 0, 1, 1,
  0, 0, 0, 1, 0, 1,
  0, 0, 0, 1, 1, 0
), byrow = TRUE, ncol = 6)
rownames(A) <- letters[1:nrow(A)]
colnames(A) <- rownames(A)

threshold_diffusion(A, seeds = c("a", "b"), threshold = 0.5)
```

`trans_coef` *Transitivity*

Description

This measure is sometimes called clustering coefficient.

Usage

```
trans_coef(
  A,
  method = c("weakcensus", "global", "mean", "local", "barrat"),
  select = c("all", "in", "out")
)
```

Arguments

`A`	A matrix
`method`	Whether to calculate the `weakcensus`, `global` transitivity ratio, the `mean` transitivity, the `local` transitivity or the weighted transitivity of `barrat`.
`select`	Whether to consider `all`, `in` or `out` ties for the local transitivity.

Value

Return a transitivity measure

Author(s)

Alejandro Espinosa-Rada

References

Barrat, A., Barthelemy, M., Pastor-Satorras, R. and Vespignani, A. (2004). The architecture of complex weighted networks. Proceedings of the National Academy of Sciences, 101(11), 3747–3752. doi:10.1073/pnas.0400087101

Wasserman, S. and Faust, K. (1994). Social network analysis: Methods and applications. Cambridge University Press.

Examples

```
A <- matrix(c(
  0, 1, 0, 1, 0,
  1, 0, 1, 1, 0,
  0, 1, 0, 0, 0,
  1, 1, 0, 0, 1,
  0, 0, 0, 1, 0
), byrow = TRUE, ncol = 5)
```

```
rownames(A) <- letters[1:ncol(A)]
colnames(A) <- rownames(A)

trans_coef(A, method = "local")

# The weighted transitivity of Barrat et al. (2004) weighs each triangle
# by the strength of the two ties of the node
W <- matrix(c(
  0, 4, 0, 2, 0,
  4, 0, 1, 3, 0,
  0, 1, 0, 0, 0,
  2, 3, 0, 0, 5,
  0, 0, 0, 5, 0
), byrow = TRUE, ncol = 5)
trans_coef(W, method = "barrat")
```

`trans_matrix` *Transitivity matrix*

Description

This function assigns a one in the elements of the matrix if a group of actors are part of a transitivity structure (030T label considering the MAN triad census)

Usage

```
trans_matrix(A, loops = FALSE)
```

Arguments

`A`	A matrix
`loops`	Whether to expect nonzero elements in the diagonal of the matrix

Value

A vector assigning an id the components that each of the nodes of the matrix belongs

Author(s)

Alejandro Espinosa-Rada

References

Davis, J.A. and Leinhardt, S. (1972). "The Structure of Positive Interpersonal Relations in Small Groups." In J. Berger (Ed.), Sociological Theories in Progress, Vol. 2, 218-251. Boston: Houghton Mifflin.

Wasserman, S. and Faust, K. (1994). Social network analysis: Methods and applications. Cambridge University Press.

Examples

```
A <- matrix(
  c(
    0, 1, 1, 0, 0, 0,
    0, 0, 1, 0, 0, 0,
    0, 0, 0, 1, 0, 0,
    0, 0, 0, 0, 0, 0,
    0, 0, 1, 1, 0, 0,
    0, 0, 0, 0, 0, 0
  ),
  byrow = TRUE, ncol = 6
)
rownames(A) <- letters[1:NROW(A)]
colnames(A) <- rownames(A)
trans_matrix(A, loops = TRUE)
```

---

traversal_weights *Traversal weights for main path analysis*

Description

Computes the Search Path Count (SPC), the Search Path Link Count (SPLC), or the Search Path Node Pair (SPNP) of each arc of a citation network (Hummon and Doreian, 1989; Batagelj, 2003), and the weighted in-degree and out-degree of each node (Kuan, 2020).

Usage

```
traversal_weights(
  A,
  method = c("spc", "splc", "spnp"),
  years = NULL,
  cutoff = NULL,
  normalized = FALSE
)
```

Arguments

A	A square, named, directed adjacency matrix in which `A[i,j] > 0` means that knowledge flows from paper `i` to paper `j` (paper `j` cites paper `i`). Only whether each arc is present is used. See dag.
method	One of `"spc"` (default), `"splc"`, or `"spnp"`.
years	Optional named numeric vector of publication years aligned with row/column names of `A`. Required when `cutoff` is supplied.
cutoff	Optional integer year. When provided together with `years`, outgoing arcs from papers published after `cutoff` are excluded from path counting, as a correction for right censoring: the most recent papers have not had time to be cited. Such papers become sinks and their outgoing arcs get a weight of zero.

`normalized` Whether to divide the weights by the total number of search paths of the method, so that the weight of an arc is the proportion of the search paths that go through it. The total is the number of paths from the sources to the sinks (SPC), from any node to the sinks (SPLC), or between any two nodes (SPNP), counting only paths with at least one arc.

Details

The weight of an arc is the number of search paths that go through it. The three weights differ in where the search paths start and end (Liu, Lu and Ho, 2019):

SPC Paths from any source to any sink. The weight of the arc $u \to v$ is the number of paths from the sources to u times the number of paths from v to the sinks.

SPLC Paths from any node to any sink, so that the intermediate papers are also origins of knowledge. The first factor becomes the number of paths from any node to u, counting u itself.

SPNP Paths from any node to any node, so that the intermediate papers are also origins and destinations of knowledge. The second factor becomes the number of paths from v to any node, counting v itself.

The three weights satisfy SPNP $\geq$ SPLC $\geq$ SPC. Liu, Lu and Ho (2019) recommend SPLC to trace the diffusion of knowledge, and Kuan (2020) argues for SPLC or SPNP.

The weighted in-degree (WiD) and out-degree (WoD) of a node are the sums of the weights of its incoming and outgoing arcs. Kuan (2020) recommends the WoD of SPLC or SPNP as the weight of a paper: it counts the influence of all the papers that precede it, and of the paper itself, on the papers that follow it, and it is zero for the sinks, whose importance is yet to be determined. The average of the two, which Kuan calls WxD, is `(weighted_indegree + weighted_outdegree) / 2`.

The numbers of paths grow exponentially with the length of the chains of citations, so they are computed in logs to avoid overflow.

Value

A named list:

`edge_weights` Square matrix of the same dimensions as `A` containing the chosen traversal weight for each arc.

`weighted_indegree` Named numeric vector with the sum of the weights of the incoming arcs of each node (WiD).

`weighted_outdegree` Named numeric vector with the sum of the weights of the outgoing arcs of each node (WoD).

`log_forward_source` Log of the number of paths from the sources to each node.

`log_forward_all` Log of the number of paths from any node to each node, counting the node itself.

`log_backward_sink` Log of the number of paths from each node to the sinks.

`log_backward_all` Log of the number of paths from each node to any node, counting the node itself.

`sources` Character vector of source node names.

`sinks` Character vector of sink node names.

`total_log_paths` Log of the total number of search paths of the method (see `normalized`).

`method` The method string used.

Author(s)

Alejandro Espinosa-Rada

References

Batagelj, V. (2003). Efficient algorithms for citation network analysis. arXiv:cs/0309023.

Hummon, N.P. and Doreian, P. (1989). Connectivity in a citation network: The development of DNA theory. Social Networks. 11(1): 39-63. doi:10.1016/03788733(89)900178.

Kuan, C. H. (2020). Regarding weight assignment algorithms of main path analysis and the conversion of arc weights to node weights. Scientometrics, 124(1), 775-782. doi:10.1007/s11192020-034688.

Liu, J.S. and Lu, L.Y.Y. (2012). An integrated approach for main path analysis: Development of the Hirsch index as an example. Journal of the American Society for Information Science and Technology. 63(3): 528-542. doi:10.1002/asi.21692.

Liu, J.S., Lu, L.Y.Y. and Ho, M.H.C. (2019). A few notes on main path analysis. Scientometrics. 119(1): 379-391. doi:10.1007/s1119201903034x.

Verspagen, B. (2007). Mapping technological trajectories as patent citation networks: A study on the history of fuel cell research. Advances in Complex Systems. 10(1): 93-115. doi:10.1142/S0219525907000945.

Examples

```
# Network of Fig. 2 in Kuan (2020)
A <- matrix(c(
 0,0,0,1,0,0,0,0,0,0,0,0,0,
 0,0,0,1,0,0,0,0,0,0,0,0,0,
 0,0,0,1,0,0,0,0,0,0,0,0,0,
 0,0,0,0,1,1,0,0,0,0,0,0,0,
 0,0,0,0,0,0,1,0,0,0,0,0,0,
 0,0,0,0,0,0,0,1,0,0,0,0,0,
 0,0,0,0,0,0,0,1,1,1,1,0,0,
 0,0,0,0,0,0,0,0,0,0,0,1,1,
 0,0,0,0,0,0,0,0,0,0,0,0,0,
 0,0,0,0,0,0,0,0,0,0,0,0,0,
 0,0,0,0,0,0,0,0,0,0,0,0,0,
 0,0,0,0,0,0,0,0,0,0,0,0,0,
 0,0,0,0,0,0,0,0,0,0,0,0,0
), nrow = 13, byrow = TRUE)

rownames(A) <- colnames(A) <- 1:13

spc <- traversal_weights(A, method = "spc")
splc <- traversal_weights(A, method = "splc")
spnp <- traversal_weights(A, method = "spnp")
```

```
# Arc 8 -> 12 has SPC 6 and SPLC 12; arc 1 -> 4 has SPNP 13 (Kuan, 2020)
spc$edge_weights["8", "12"]
splc$edge_weights["8", "12"]
spnp$edge_weights["1", "4"]

# Table 4 in Kuan (2020)
cbind(
  SPLC_WoD = splc$weighted_outdegree,
  SPNP_WoD = spnp$weighted_outdegree,
  SPNP_WxD = (spnp$weighted_indegree + spnp$weighted_outdegree) / 2
)
```

`triad_uman` *Triad census analysis assuming U\|MAN*

Description

Considering the triad census of Davis and Leinhardt (1972) for vector A, B, and C:

Usage

```
triad_uman(A, ztest = FALSE, covar = FALSE, l = NULL)
```

Arguments

`A`	A symmetric matrix object
`ztest`	Return Z and p-value
`covar`	Return the covariance matrix for triadic analysis
`l`	Optional vector with the sixteen coefficients of a linear combination of the triad census (e.g. a transitivity weighting vector). The observed value, its expectation, standard deviation and a z-test are returned

Details

003 = A,B,C, empty triad

012 = A -> B, C, triad with a single directed edge

102 = A <-> B, C, triad with a reciprocated connection between two vertices

021D = A <-B-> C, triadic out-star

021U = A -> B <- C triadic in-star

021C = A-> B-> C, directed line

111D = A <-> B <-C

111U = A <-> B-> C

030T = A-> B <-C, A-> C

030C = A <-B <-C, A-> C

201 = A <-> B <-> C

120D = A <-B-> C, A <-> C

120U = A-> B <-C, A <->C

120C = A-> B-> C, A <-> C

210 = A-> B <-> C, A <-> C

300 = A <-> B <-> C, A <->C, complete triad.

Value

This function gives the counts of the triad census, the expected counts, assuming that U|MAN distribution (Holland and Leinhardt, 1975, 1976) is operating, and the standard deviations of these counts. The covariance matrix and the test of a linear combination of the census are returned when requested.

Author(s)

Alejandro Espinosa-Rada

References

Davis, J.A. and Leinhardt, S. (1972). The Structure of Positive Interpersonal Relations in Small Groups. In J. Berger (Ed.), Sociological Theories in Progress, Volume 2, 218-251. Boston: Houghton Mifflin.

Holland, P. W. and Leinhardt, S. (1975). The statistical analysis of local structure in social networks. In D. R. Heise (Ed.), Sociological Methodology, 1976 (Jossey-Bass, pp. 1–45).

Holland, P. W. and Leinhardt, S. (1976). Local Structure in Social Networks. Sociological Methodology, 7, 1–45. doi:10.2307/270703

Wasserman, S. and Faust, K. (1994). Social network analysis: Methods and applications. Cambridge University Press.

Examples

```
data(krackhardt_friends)
triad_uman(krackhardt_friends)

triad_uman(krackhardt_friends, ztest = TRUE, covar = TRUE)

# Transitivity: 030T, 120D, 120U and 300 are transitive triads
l <- c(0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 1, 0, 0, 1)
triad_uman(krackhardt_friends, l = l)$z_test
```

z_arctest *Z test of the number of arcs*

Description

Z test of the number of arcs

Usage

```
z_arctest(G, p = 0.5, interval = FALSE)
```

Arguments

G	A symmetric matrix object.
p	Constant probability p.
interval	Return a 95 percent confidence interval.

Value

This function gives a Z test and p-value for the number of lines or arcs present in a directed graph

Author(s)

Alejandro Espinosa-Rada

References

Wasserman, S. and Faust, K. (1994). Social network analysis: Methods and applications. Cambridge University Press.

Examples

```
data(krackhardt_friends)
z_arctest(krackhardt_friends)
```

zone_sample *Zone-2 sampling from second-mode*

Description

Second-zone multilevel sampling considering a second-mode focal actor

Usage

```
zone_sample(A, X, ego = TRUE, core = FALSE)
```

Arguments

A	A symmetric matrix object.
X	X an incidence matrix object.
ego	Whether to add or not ego into the subgraph.
core	Whether to add actors at distance one from ego

Value

This function return a list of second-zone subgraphs using as a focal actor the second-mode of the multilevel network. Each subgraph is the binary adjacency matrix of the meta-matrix of A and X restricted to the nodes of the zone, without loops. When core = TRUE, each matrix has an attribute core, a named vector that is one for the actors at distance one from the focal node and zero otherwise.

Author(s)

Alejandro Espinosa-Rada

References

Espinosa-Rada, A. (2021). A Network Approach for the Sociological Study of Science: Modelling Dynamic Multilevel Networks. [PhD](https://research.manchester.ac.uk/en/studentTheses/a-network-approach-for-the-sociological-study-of-science-and-know). The University of Manchester.

Examples

```
A <- matrix(c(
  0, 1, 0, 0, 0, 0, 0, 0,
  0, 0, 1, 0, 0, 0, 0, 0,
  0, 1, 0, 1, 0, 0, 0, 0,
  0, 0, 0, 0, 0, 0, 0, 0,
  0, 0, 0, 0, 0, 0, 0, 0,
  0, 0, 0, 1, 0, 0, 0, 0,
  0, 0, 0, 0, 0, 0, 0, 0,
  0, 0, 0, 0, 0, 0, 0, 0
), byrow = TRUE, ncol = 8)
colnames(A) <- c("1", "2", "3", "4", "5", "6", "7", "8")
rownames(A) <- c("1", "2", "3", "4", "5", "6", "7", "8")

X <- matrix(c(
  1, 0, 0, 0,
  1, 0, 0, 0,
  1, 0, 1, 0,
  0, 1, 1, 0,
  0, 1, 1, 1,
  0, 1, 0, 0,
  0, 0, 0, 0,
  0, 0, 0, 1
), byrow = TRUE, ncol = 4)
colnames(X) <- c("a", "b", "c", "d")
```

```
rownames(X) <- c("1", "2", "3", "4", "5", "6", "7", "8")

set.seed(18051889)
zone_sample(A, X, core = TRUE)
```

Index

∗ **datasets**
  campnet, 19
  FIFAego, 60
  FIFAex, 61
  FIFAin, 61
  krackhardt_friends, 82
  lazega_lawfirm, 84

adj_to_incidence, 4
adj_to_matrix, 5
aggregate_layers, 6
alter_composition, 8
alter_heterogeneity, 9
alter_homophily, 10

betweenness_centrality, 12
bfs_ugraph *(*distances*)*, 43
block_density, 13
bonacich_norm, 15
bonacich_power, 16
brokerage_roles, 17

campnet, 19
centrality_centralization, 20
citation_decay, 21
clique_max, 22
clique_table, 23
closeness_centrality, 25
co_occurrence, 26
communities, 27
community_betweenness *(*communities*)*, 27
community_greedy *(*communities*)*, 27
community_label *(*communities*)*, 27
components_id, 29
compound_relation, 30
concor, 32
core_periphery, 33
count_geodesics *(*distances*)*, 43
cug_test, 34
cumulativeSumMatrices, 36

dag, 37, *88*, *149*
dag_check *(*dag*)*, 37
dag_sort *(*dag*)*, 37
dir_inclusion, 39
dist_geographic, 40
dist_sim_matrix, 41
distances, 43
dominance_layers, 45
dominance_pairs, 47
dominance_ranks, 48
dyad_triad_table, 49
dyadic_census, 50

eb_constraint, 51
edgelist_to_matrix, 53
ego_net, 54
ei_index, 56
eigenvector_centrality, 57
expand_matrix, 58
extract_component, 59

FIFAego, 60
FIFAex, 61
FIFAin, 61
fractional_approach, 62

gen_degree, 64
gen_density, 65
geo_distances *(*geodesics*)*, 67
geo_summary *(*geodesics*)*, 67
geodesics, 67

heterogeneity, 69
hyperevent_dominance, 70
hypergraph, 72

ind_rand_matrix, 73
indirect_rel, 75

jaccard, 76

k_core, 78
katz_centrality, 80
kp_reciprocity, 81
krackhardt_friends, 82
krackhardt_index, 82

lazega_lawfirm, 84
leading_eigen, 85
leiden, 86

main_path, 87
main_path_diag, 89
matrix_adjlist, 91
matrix_projection, 92
matrix_report, 93
matrix_to_edgelist, 94
meta_matrix, 95
minmax_overlap, 97
mix_matrix, 98
mixed_census, 99
modularity_score, 100
multilevel_degree, 102
multiplex_census, 105

neigh_inclusion, 107

page_rank_centrality, 109
pareto_dominance, 110
partition_centrality, 112
percolation_clique, 113
perm_label, 114
perm_matrix, 115
pos_dominance, 116
posneg_index, 117
power_function, 118
pref_attachment, 119
preserved_order, 120

q_analysis, 121
qap_cor, 124
qap_lm, 126

recip_coef, 127
redundancy, 128
rege, 129

segregation, 130
set_inclusion, 132
shared_partners, 133
short_path (distances), 43
simplicial_complexes, 134
small_world, 136
social_influence, 137
spatial_cor, 139
struc_balance, 140
structural_holes, 141
structural_na, 143
supra_adjacency, 144

threshold_diffusion, 145
trans_coef, 147
trans_matrix, 148
traversal_weights, *21*, *22*, 149
triad_uman, 152

wall_distances (distances), 43
wlocal_distances (distances), 43

z_arctest, 154
zone_sample, 154